IN THIS ISSUE

June 8, 2021

A. percula in Kimbe Bay. Image credit: Tane Sinclair-Taylor (photographer).

DEVELOPMENTAL BIOLOGY

How clownfish gain their stripes

Charismatic denizens of coral reefs, clownfish are easily spotted by the sinuous white bars that lend their streamlined bodies a striped appearance. The white bars develop during metamorphosis, but the driving factors remain unclear. Pauline Salis et al. report that the rate of white bar formation in the clownfish *Amphiprion percula*, which lives in symbiosis with two different sea anemone hosts in the tropical waters of Kimbe Bay, Papua New Guinea, varies depending on the host species. New recruits of *A. percula* found in the anemone host *Stichodactyla gigantea* exhibited more white bars than those of similar age and size found in the anemone host *Heteractis magnifica*. Experiments in which larval clownfish were exposed to varying concentrations of thyroid hormone (TH) revealed that TH acts on color-producing cells called iridophores and influences white bar development in a dose-dependent manner; blocking TH delayed white bar formation. Comparison of gene expression in *A. percula* recruits in the two hosts uncovered a role for the *duox* gene, which encodes the enzyme dual oxidase, implicated in TH production; *duox* was overexpressed in new *S. gigantea* recruits, compared with new *H. magnifica* recruits. Although the ecological significance of white bar formation remains unclear, the findings raise the possibility that TH-driven bar formation in clownfish may be tied to their differential recruitment in and adjustment to different sea anemone species, according to the authors. — P.N.

Read online

POLITICAL SCIENCES

Machine learning and satellite imagery monitor war destruction

Estimating building destruction in urban war zones is challenging, and attempts to use satellite imagery and artificial intelligence have failed to achieve high precision in real-world applications. Hannes Mueller et al. developed an automated method to measure building destruction in publicly available high-resolution satellite images. The authors used deep learning to spot destruction features from heavy weaponry attacks and used images of surrounding areas and multiple successive images of the same area to reduce error rates. Next, the authors applied the method to repeated satellite images of the entire populated areas of major Syrian cities between 2011 and 2017 to produce longitudinal estimates of building destruction over the course of the Syrian civil war. The method achieved an area under the curve of above 0.9 and an average precision of over 0.42 in the unbalanced sample from six Syrian cities and displayed high performance in identifying the timing and location of building destruction in parts of Aleppo that were not part of the training sample. According to the authors, the method enables the automated classification of building destruction from repeated high-resolution satellite imagery and could be applied to other populated areas with near-real-time tracking. — S.R.

Read online

Monitoring war destruction from space using machine learning

Hannes Mueller[a,b,1], Andre Groeger[b,c,1], Jonathan Hersh[d], Andrea Matranga[d,e], and Joan Serrat[f,g]

[a]Institute of Economic Analysis, Spanish National Research Council (CSIC), 08193 Bellaterra, Spain; [b]Barcelona Graduate School of Economics, 08005 Barcelona, Spain; [c]Department of Economics and Economic History, Universitat Autònoma de Barcelona, 08193 Bellaterra, Spain; [d]Argyros School of Business, Chapman University, Orange, CA 92868; [e]Smith Institute for Political Economy and Philosophy, Chapman University, Orange, CA 92868; [f]Computer Science Department, Universitat Autònoma de Barcelona, 08193 Bellaterra, Spain; and [g]Computer Vision Center, Universitat Autònoma de Barcelona, 08193 Bellaterra, Spain

Edited by Douglas S. Massey, Princeton University, Princeton, NJ, and approved April 26, 2021 (received for review December 11, 2020)

Existing data on building destruction in conflict zones rely on eyewitness reports or manual detection, which makes it generally scarce, incomplete, and potentially biased. This lack of reliable data imposes severe limitations for media reporting, humanitarian relief efforts, human-rights monitoring, reconstruction initiatives, and academic studies of violent conflict. This article introduces an automated method of measuring destruction in high-resolution satellite images using deep-learning techniques combined with label augmentation and spatial and temporal smoothing, which exploit the underlying spatial and temporal structure of destruction. As a proof of concept, we apply this method to the Syrian civil war and reconstruct the evolution of damage in major cities across the country. Our approach allows generating destruction data with unprecedented scope, resolution, and frequency—and makes use of the ever-higher frequency at which satellite imagery becomes available.

conflict | destruction | deep learning | remote sensing | Syria

Building destruction during war is a specific form of violence that is particularly harmful to civilians, commonly used to displace populations, and therefore warrants special attention. Yet, data from war-ridden areas are typically scarce, often incomplete, and highly contested, when available. The lack of such data from conflict zones severely limits media reporting, humanitarian relief efforts, human-rights monitoring, and reconstruction initiatives, as well as the study of violent conflict in academic research. A novel solution to this problem is to use remote sensing to identify destruction in satellite images (1–3). This approach is gaining momentum as high-resolution imagery is becoming readily available at ever-higher frequency, yielding weekly, or even daily, images. At the same time, recent methodological advances related to deep learning have provided sophisticated tools to extract data from these images (4–7).

While seminal research has demonstrated the use of automated classifiers for destruction detection, practical applications have so far been hampered by severe problems with labeling, domain transfer, and class imbalance in real-world imagery from urban war zones. As a consequence, international organizations such as the United Nations, the World Bank, and Amnesty International use remote sensing with manual human classification to produce damage-assessment case studies (8–10). On the other hand, providers of conflict data for research purposes still rely heavily on news and eyewitness reports, which leads to large data-publishing lags and potential biases (11–17). An automated building-damage classifier for use with satellite imagery, which has a low rate of false positives in unbalanced samples and allows tracking on-the-ground destruction in close to real-time, would therefore be extremely valuable for the international community and academic researchers alike.

In this article, we present a way of combining computer-vision techniques and publicly available high-resolution satellite images to produce building-destruction estimates that are of practical use to both practitioners and researchers. The standard architectures for this task are convolutional neural networks (CNNs),* as they have achieved unprecedented success in large-scale visual image classification with error rates beating humans (18, 19). We train a CNN to spot destruction features from heavy weaponry attacks (i.e., artillery and bombing) in satellite images, such as the rubble from collapsed buildings or the presence of bomb craters.

We make three relevant methodological contributions. First, we introduce a label-augmentation method for expanding destruction class labels by making reasonable assumptions about the data-generating process using contextual information. Second, we introduce a two-stage classification process to control for spatial and temporal noise where the results from the CNN are processed through a random-forest model that relies on spatial and temporal leads and lags to improve classification performance. Third, we apply our trained computer-vision model to repeated satellite images of the entire populated areas of major Syrian cities, including parks and highways, and produce longitudinal estimates of building destruction over the course of the recent civil war.

We demonstrate that our method yields high performance in out-of-sample tests and validate its ability for destruction

Significance

Satellite imagery is becoming ubiquitous. Research has demonstrated that artificial intelligence applied to satellite imagery holds promise for automated detection of war-related building destruction. While these results are promising, monitoring in real-world applications requires high precision, especially when destruction is sparse and detecting destroyed buildings is equivalent to looking for a needle in a haystack. We demonstrate that exploiting the persistent nature of building destruction can substantially improve the training of automated destruction monitoring. We also propose an additional machine-learning stage that leverages images of surrounding areas and multiple successive images of the same area, which further improves detection significantly. This will allow real-world applications, and we illustrate this in the context of the Syrian civil war.

Author contributions: H.M., A.G., J.H., A.M., and J.S. designed research; H.M., A.G., J.H., and J.S. performed research; H.M. and A.G. analyzed data; and H.M., A.G., J.H., A.M., and J.S. wrote the paper.

The authors declare no competing interest.

This article is a PNAS Direct Submission.

[1]To whom correspondence may be addressed. Email: andre.groger@uab.es or h.mueller.uni@gmail.com.

This article contains supporting information online at https://www.pnas.org/lookup/suppl/doi:10.1073/pnas.2025400118/-/DCSupplemental.

Published June 3, 2021.

*For a glossary of technical key terminology, see *SI Appendix*.

monitoring using a separate database of heavy weaponry attacks. Our results highlight the importance of repeated satellite imagery in combination with temporal filtering to improve monitoring performance. As a result, our approach can be applied to any populated area, provided that repeated, high-resolution (i.e., submeter) satellite imagery is available.

Why Automated War-Destruction Monitoring Is Hard

Several studies have demonstrated the use of computer vision on satellite imagery to identify different types of destruction (2, 3, 20–26). In many cases, this is destruction from natural disasters, which tends to be spatially concentrated. While performance results from the literature are encouraging, they typically focus on evaluations at one point in time and training/validating on datasets composed of equal numbers of damaged and undamaged images.

Precision performance in repeated destruction scans of entire cities with heavily unbalanced classes, as in our application, have not been explicitly presented in the literature so far. Part of the reason for this gap is that automated methods need to be able to detect building destruction in an empirical context, where the vast majority of images do not feature destruction. Class imbalance is a common problem in machine-learning applications, but the detection of destruction in war zones faces an extreme level of imbalance. Even in a city that suffered as much destruction as Aleppo, only 2.8% of all images of populated areas contain a building that was classified as destroyed by the United Nations Operational Satellite Applications Program (UNOSAT) in September 2016.

Fig. 1 depicts this quite clearly. In Fig. 1*A*, we see the full extent of Aleppo, with all destroyed building annotations depicted as red dots. Fig. 1*B* zooms into the central area of Aleppo, just east of the historic Citadel, which was heavily attacked. The red dots coincide clearly with patterns of destruction from heavy weaponry attacks in the satellite images. But destruction only affected a small fraction of buildings, even in this heavily affected part of the city.

With such class imbalance, even a small false positive rate (FPR) will result in an unacceptable absolute number of false-positive predictions in applications, which would yield destruction data that are practically useless due to high measurement error. A simple example illustrates this: Suppose we have 100,000 sample images, of which 1,000 are destroyed. A "low" FPR of 15% together with a true positive rate (TPR) of 90% implies that the classification model will produce 14,850 false positives and 900 true positives, resulting in a precision below 6%. In other words, conditional on predicting destruction, such a classifier would be wrong more than 94% of the time. Note that the same classifier produces a "high" precision score of 86% on a 1:1 balanced sample.

The task of automated monitoring over time is typically further complicated by a lack of training data, i.e., the low number of destruction labels available in any given city. This can quickly lead to overfitting in machine learning, as the training set consists of a narrow selection of building types, neighborhoods, sun and satellite angles, and changing vegetation or weather phenomena like snow and cloud coverage. These problems are known as spatial and temporal domain shift (27). Temporal domain shift is a particularly serious problem in our application, as destruction monitoring requires the generation of a reasonable timeline with repeated scans of the same city. This emphasizes the need for a robust solution to this problem that ensures some comparability across time.

Our approach aims at solving these problems. We exploit the time dimension of the images and labels to alleviate the domain-shift problems and extreme class imbalance. We also make a point of reporting precision performance in unbalanced samples to provide realistic insights into the potential performance in applications.

Methods

Satellite Data. Most of our sample comes from Aleppo, which we use as our main proof-of-concept due to the size of the city and the high availability of repeated images and labels. To train and evaluate our model, we used 22 high-resolution satellite images from Aleppo and a total of 42 images from five other Syrian cities (Table 1). All images used in this analysis were obtained from Google Earth (28); were georeferenced and orthorectified; and feature three bands (red, green, blue), as well as a ground sampling distance of circa 50 cm per pixel.

Sample images cover the period 2011 to 2017, after the onset of the civil war in Syria, during which extensive destruction from heavy weaponry attacks occurred across all sample cities. We used an additional, early image for each city (for example, June 26, 2011, in Aleppo) as the "pre" image and call the later 64 images as the "post" images. Our method relies on change detection—i.e., when classifying images, the preimage is compared to the respective postimage.

To move as close as possible to the automated monitoring task, we transformed all images into millions of 64 × 64 pixel subimages that we call *patches*. These patches are the unit of observation for training and testing and the final step, which we call *scanning* or *dense prediction*, in which the classifier is used to produce fitted values for every patch in the study areas. Ground area coverage of each patch can vary slightly, but is approximately 1,024 (i.e., 32 × 32) square meters. Importantly, the size of a specific patch remains constant over time.

Column (2) in Table 1 reports the sample size in terms of patches for the six cities in our sample. For Aleppo, for example, we have over 95,000 patches per image times 22 images, which gives approximately 2.1 million patches. Importantly, this is panel data, where images of the same patch are repeated 22 times.

Destruction Labels. We combine the imagery data with georeferenced building damage labels from the UNOSAT of the United Nations Institute for Training and Research (UNITAR) (8). Over the course of the Syrian civil war, UNOSAT produced building-destruction annotations by manual inspection of satellite images for severely affected Syrian cities. For Aleppo, these manual assessments were conducted at four different dates, one each year between 2013 and 2016. Column (3) in Table 1 reports the number of these assessments.

UNOSAT damage annotations were categorized into three degrees of damage: moderate and severe damage, as well as completely destroyed. In our analysis, we rely on the latter class due to the fact that destruction patterns for the other labels were not always clearly visible in the satellite images. Our method classifies the satellite images as destroyed if at least one UNOSAT annotation of destruction is inside a patch.

Our analysis of building destruction focuses on the urban areas of Syrian cities. For Aleppo, this is depicted by the area enclosed by the yellow line in Fig. 1. Areas enclosed by magenta lines correspond to so-called "no analysis" areas, which have been left out by UNOSAT in their damage annotations due to these zones hosting noncivilian buildings. Consequently, these areas are also excluded from the training process. But we scan these areas and make use of these scans for out-of-sample validation. Sample image patches for destroyed areas predestruction and postdestruction are presented in *SI Appendix*, Fig. S1, and nondestroyed ones, including damaged buildings, are shown in *SI Appendix*, Fig. S2.

The ideal annotation dataset to analyze this problem would be composed of pixel-wise classification of all damaged and nondamaged buildings across the sample cities for all time periods. Labels like this could then be used to train models to identify the footprint of destroyed buildings using satellite images (3, 22). However, because of the significant cost of annotating destruction footprints, UNOSAT only provides point coordinates (centroids) of destroyed buildings. We matched these point labels to our image patches by attributing a label to the closest patch centroid. One issue with this method of generating labels is that buildings have different sizes, and, therefore, some UNOSAT labels are surrounded by more visible destruction than others. We address this issue through a second stage, described below, in which we exploit spatial information.

Contextual Label Augmentation and Test Sample. The computer-vision task is to train an algorithm to detect destruction from the visual bands of high-resolution daylight satellite images. Training deep-learning architectures

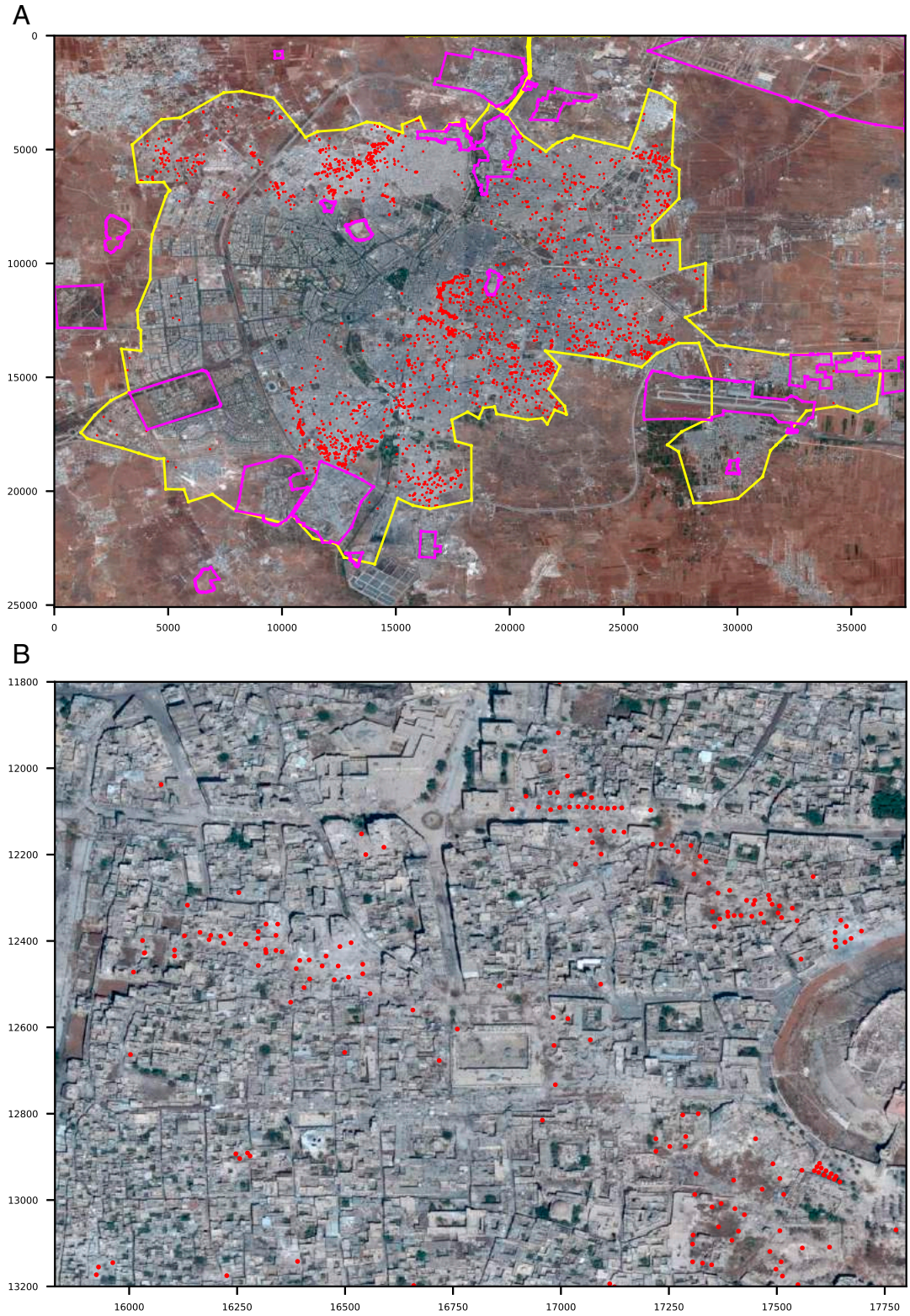


Fig. 1. Imagery of Aleppo on September 18, 2016. Red dots indicate UNOSAT annotations as destroyed. Areas enclosed by magenta lines are no analysis zones, excluded from the UNOSAT damage assessment due to being noncivilian. The yellow line encloses the populated areas of Aleppo under analysis. Sources: Google Earth/Maxar satellite imagery and UNITAR/UNOSAT damage annotations. *A* shows an overview of the urban area of Aleppo. *B* shows an area in central Aleppo close to the Citadel.

typically requires large training datasets, including thousands of labels, which are extremely rare in our empirical context.

Consequently, as reported in column (3) of Table 1, we have a maximum of four UNOSAT annotation dates to work with for certain cities, and for others, three or only two (i.e., Homs). Compared to the number of annotations, we usually have significantly more raw images available, as shown in column (1). In addition, few label dates perfectly coincide with the date of a satellite image. This generates an "uncertain class," in which patches cannot

Table 1. Sample overview

City	(1) Total images	(2) Total patches	(3) Total labeled images	(4) Total labeled patches	(5) Share destroyed patches, %
Aleppo	22	2,106,412	4	1,626,920	1.83
Daraa	13	202,462	4	125,231	1.00
Deir-Ez-Zor	7	98,602	4	84,723	2.86
Hama	9	285,057	3	224,365	3.73
Homs	5	200,035	2	83,941	8.26
Raqqa	8	180,184	3	112,481	1.96
All	64	3,072,752	20	2,257,661	2.26

Note: Column (1) reports the number of "post" satellite images/time periods, excluding the first preimage for each city. Column (2) reports the resulting number of patches in the populated areas of the respective city based on available imagery. Column (3) refers to the number of images/time periods for which UNITAR/UNOSAT labels are available. Column (4) is the number of patches for which UNITAR/UNOSAT damage labels for the "destroyed" class are available after label augmentation. Column (5) is the share of destroyed labels over the number of labeled patches. Sources: Author calculations based on Google Earth/Maxar satellite imagery and UNITAR/UNOSAT damage annotations.

be attributed clearly to either the destroyed or not-destroyed class because destruction could have occurred between the labeling date and the date of the image.

To increase the number of labeled data points, we exploit the fact that reconstruction was largely absent in the areas of interest during the study period between 2013 and 2017 (*SI Appendix*, Table S4). Our label-augmentation approach assumes that positive samples at time t_i also remained positives at subsequent times $t_j > t_i$, i.e., that destruction persists throughout the period of the civil war. And, conversely, that negative samples at time t_j also had to be negatives at times $t_i < t_j$.

We solve two problems using this approach. First, we expand the size of our training dataset by boosting the number of labels to close to 2.3 million, of which approximately 51,000 show destruction. Second, by including additional time periods in our training sample, we improve the performance of our classifier in its ability to handle domain shift. Our method of label augmentation is conservative, given that we assign missing values to all patches that remain in the uncertain class—those for which we cannot know with certainty whether destruction has occurred in the past or those for which we do not know with certainty that they will be labeled not destroyed in the future.

Fig. 2 illustrates our method for generating training and test samples. Given the temporal and spatial structure of the data, extra care must be taken when splitting the sample for training and testing to avoid overfitting. Standard cross-sectional cross-validation procedures are not appropriate since they could show the network patches from different times, but the same location, in training and testing. We therefore used the patch identifier to perform sample splitting, whereby 70% of patches are reserved for training and 30% for testing across temporal periods. All performance measures reflect accuracy as measured from data reserved in the test set.

CNN Architecture and Two-Stage Classification Procedure. Another innovation in our approach is the use of a two-stage classification procedure that feeds the predicted destruction estimates from the initial CNN model into a random-forest classifier. With respect to the CNN architecture, we experimented with several different types of CNNs. For each of these, we optimized hyperparameters according to accuracy results in the validation set. The results of these experiments suggested the use of a relatively flat CNN architecture, as described in *SI Appendix*, section 1.

To the output of the CNN model, we applied a second machine-learning stage, intended to exploit the temporal and spatial clustering of destruction. Specifically, the labels and predicted values from the CNN are used to train a random-forest model that relies on information from two spatial lags around each patch location and two temporal leads/lags around each date. The random forest uses these spatial and temporal features from the raw CNN scores plus the spatial SD to generate a prediction for the test sample and the dense prediction.

The logic behind this second-stage approach is that destruction is not only serially correlated, but also spatially clustered. We separated this step from the deep-learning stage for maximum flexibility and modularity. This allows us to vary the information set that we used in the second-stage model. In particular, we experimented with using only spatial information and different temporal lag structures and discuss their relative importance below.

Data Generation. As a final step, we trained the second stage on all available data and predicted values for every patch-period combination in our data. This simulates the data-generation problem where the trained architecture is used to interpret all patches at all points in time, including those patches that had missing labels. The result is what we call *dense* predictions, and this forms the raw material for additional validation exercises. As reported in column (2) of Table 1, the result is a panel dataset of destruction predictions at the patch level for six cities with varying time periods with over 3 million patch-time observations.

Results

Overall Performance. Our first-stage CNN classifier achieves an area under the curve (AUC) of 0.86 in the test sample of the first stage (i.e., with the raw output from the CNN) and an

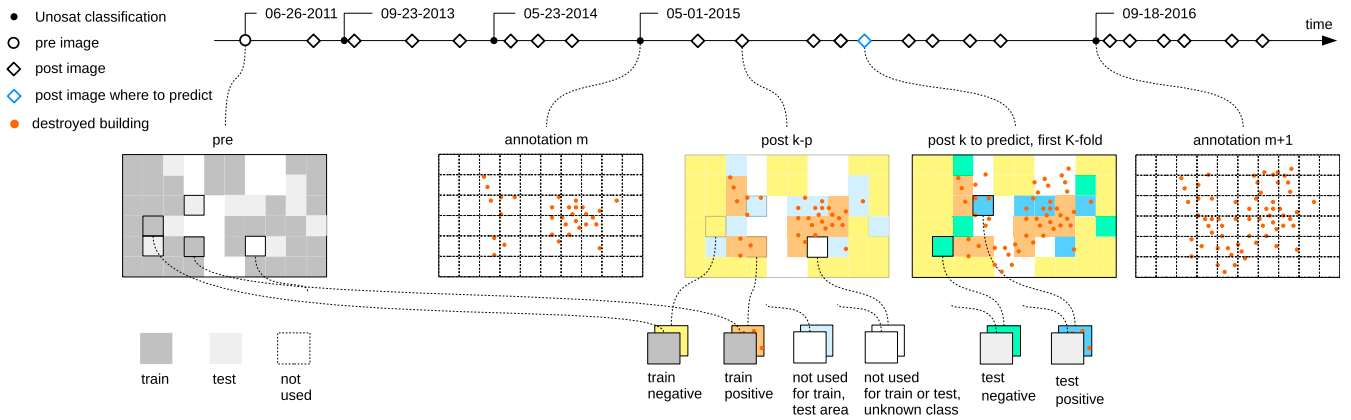


Fig. 2. Image sampling and prediction process. The timeline shows 23 Aleppo images. The first image, from June 26,2011, is used as a prewar image when training the classifier. All the other 22 images are used as postimages. Images are split into over 95,000 patches, which serve as a unit of analysis and are separated into test and training sample before the analysis. Labels for the patches come from UNITAR/UNOSAT annotation dates shown as black dots on the timeline. Annotations are extended forward and backward in time beyond these dates under the assumption that buildings that are labeled destroyed at some point remain destroyed throughout the period of observation. Those that are labeled as not destroyed at a given time were not destroyed before. Patches that are not destroyed at an annotation date, but are destroyed at a later annotation date, have an unknown class. All patches that are not classified as destroyed in the last annotation date are of unknown class (set to missing) after that date.

AUC of 0.92 after the second-stage random-forest procedure (*SI Appendix*, Fig. S4). The associated receiver operating characteristic (ROC) curve implies that a TPR of 0.8 is associated with an FPR of 0.17. At a more conservative, higher threshold for a positive classification, a TPR of 0.5 is associated with an FPR of only 0.025. However, the class imbalance is extremely relevant here. The ROC curve and its AUC are classification performance measures that are not affected by class imbalance in the sample and, therefore, do not allow us to discuss the impact of class imbalance in our sample. In what follows, we, therefore, focus on precision statistics to highlight the problem of unbalanced classes in applications of automated destruction detection.

Fig. 3 summarizes our main results across cities. Fig. 3*A* presents two precision-recall curves from the test sample that depict the out-of-sample performance of our classification approach. The dashed orange curve plots the precision-recall trade-off in the balanced sample. The average precision here is 0.86, and the curve suggests a very mild trade-off with a precision of over 0.9 at a recall rate of 0.5, for example. In contrast, the solid blue line depicts the performance of the same model when taking into account unbalanced classes that the automated destruction detection would face in the actual application in the test sample. Clearly, precision is much lower with the average precision being a mere 0.24. For a recall rate of 0.5, the first stage reaches a precision of below 0.2. This illustrates

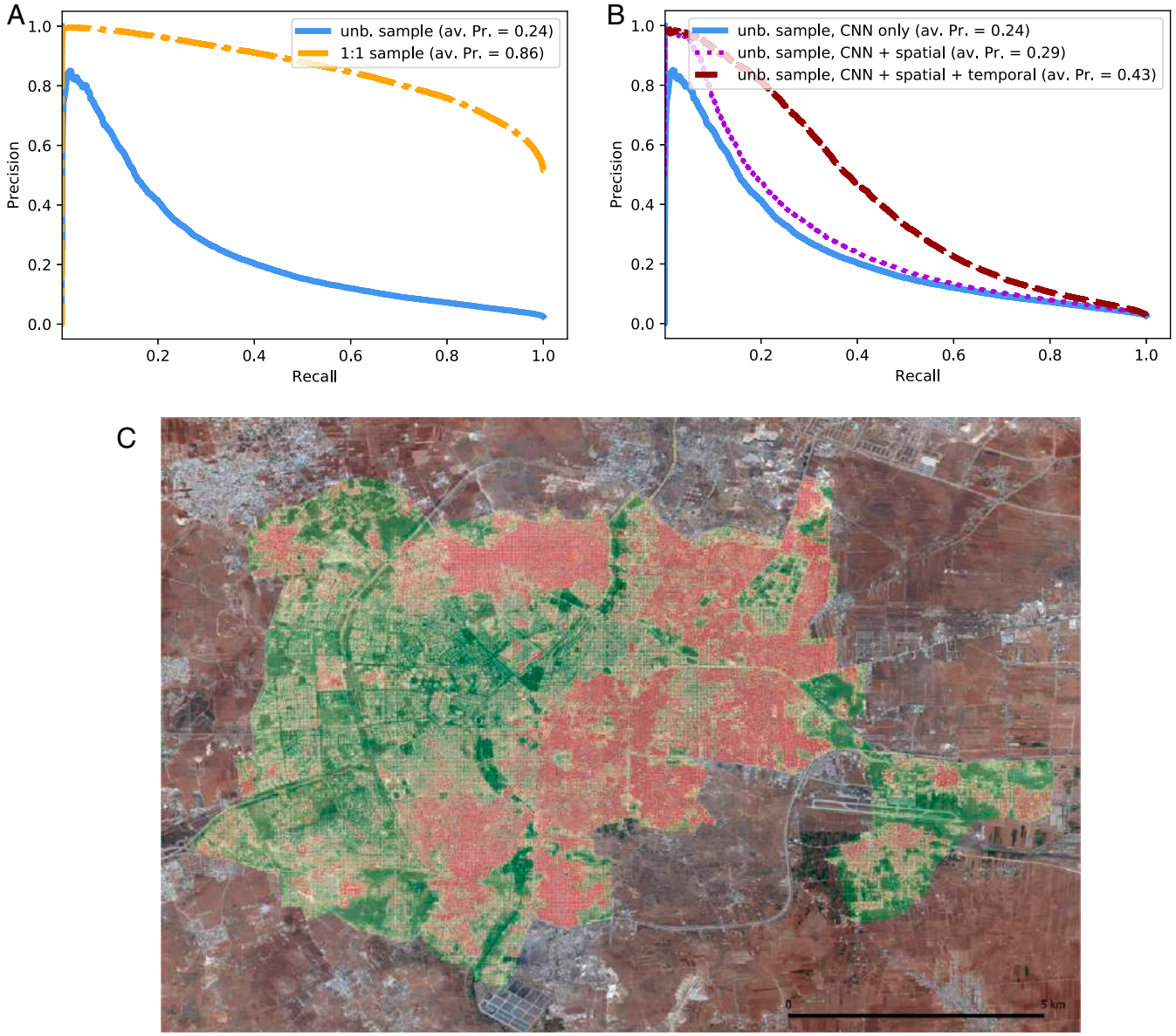


Fig. 3. (*A*) Precision-recall curve, unbalanced versus balanced sample. Reported performance is in the 30% training sample either by up-sampling the positives to reach a 1:1 sample (orange curve) or by evaluating at the original sample proportions (blue curve). (*B*) Precision-recall curve, unbalanced sample. First-stage model versus two alternative second-stage models. As in *A*, the blue curve shows performance after the first stage. The dashed maroon curve shows performance after the second stage, which uses training of a random forest on temporal and spatial leads and lags in the training sample. The dotted purple curve shows performance when using only spatial lags and no additional temporal information. (*C*) Average second-stage dense patch-wise destruction-prediction scores for Aleppo city, Syria. Green color indicates low prediction scores, and red color indicates high prediction scores. Color bins reflect deciles of second-stage fitted values with full spatial and temporal smoothing. Sources: Google Earth/Maxar satellite imagery, UNITAR/UNOSAT damage annotations, and author calculations.

impressively how class imbalance in real application can change the precision-recall trade-off in this exercise.

In Fig. 3*B*, we illustrate the improvement in precision that we achieve by applying the second stage. The figure compares precision-recall curves for the first stage (solid blue line), as in Fig. 3*A*, with the improvements from the second-stage models, all evaluated in the unbalanced test sample. The second-stage average precision increased to 0.29 with only spatial smoothing (dotted purple line) and 0.43 with temporal and spatial smoothing (dashed maroon line). This highlights a key insight from our experiments with the modular second stage. The use of temporal smoothing is absolutely crucial for reaching better precision in the second stage. The gains of the spatial smoothing are relevant in some cases, but the real boost in performance arises when using temporal information to validate predictions coming out of the first stage.

In Fig. 3*C*, we show an example of the final output of our methodology—the continuous dense prediction scores generated from the second stage. The figure shows the average patch-wise dense predictions across the entire city of Aleppo, including no-analysis zones. Red color indicates high predicted scores, and green indicates low scores. Generally, the red areas coincide with the destruction annotations in Fig. 1. In addition, roads and parks are clearly visible as dark green (lowest destruction probability) or yellow patches. This is not only evidence of the power of our approach in picking up housing destruction, but it also shows how the classifier has learned that roads and parks are never destroyed buildings.

The Role of the Second-Stage Module. The second stage plays a key role for boosting performance to levels that imply practical gains from automatizing destruction monitoring in our sample. It is important to consider that, while the cities in our sample are all in the same country, they are of different sizes, have different building types, and are situated in different landscapes with a variety of vegetation and seasonal changes. In addition, label and image availability differ dramatically. As shown in Table 1, the vast majority of images in our sample come from Aleppo due to its large size and elevated image availability—less than one-third of all images come from other cities (*SI Appendix*, Table S3 summarizes the results from training on Aleppo exclusively). If our approach can adapt to these very different conditions, it means we can be optimistic about applications elsewhere.

Table 2. Model performance when varying second-stage module in the unbalanced sample

	(1) First-stage (CNN)	(2)	(3) Second-stage (CNN + RF)	(4)
City	Raw Precision	With spatial leads/lags Precision	With spatial and temporal leads/lags Precision	With spatial and temporal leads/lags AUC
Aleppo	16.1	16.9	35.7	91.5
Daraa	4.2	4.6	11.7	89.0
Deir-Ez-Zor	11.0	12.1	21.7	80.0
Hama	54.5	65.2	68.0	91.0
Homs	25.8	34.9	55.2	85.7
Raqqa	12.8	17.4	32.1	87.6
All	24.5	28.7	42.5	90.7

Notes. First-stage predictions from CNNs and second-stage predictions from random-forest model (CNN + RF) with spatial leads/lags (column 2) and spatial and two temporal leads/lags (columns 3 and 4). Columns (1)–(3) report the average precision and column (4) the AUC. Sources: Author calculations based on Google Earth/Maxar satellite imagery and UNITAR/UNOSAT damage annotations.

Table 2 provides details on the performance improvements through the second-stage procedure by city. In column (1), we report performance of the first stage by city. This reveals strong differences in performance across cities, with average precision ranging from a mere 4.2% for Daraa to an impressive 54.5% for Hama (for corresponding precision-recall curves, see *SI Appendix*, Fig. S5). To a large degree, this is driven by sample imbalances, where Daraa suffered only 1% of destroyed patches, on average, whereas Hama suffered almost four times as much.

The second stage boosts this performance substantially. This is most notable for the worst-performing cities, for which precision improves twofold to threefold in the full model (column 3). How does the full model achieve this improvement in performance? Table 2 confirms the role of the temporal smoothing shown in Fig. 3. However, the city-by-city analysis also reveals interesting differences across cities, where Homs and Hama seem to benefit more from the spatial smoothing. In both cities, destruction is indeed clustered heavily in some neighborhoods, so that this clustering might be useful in reinforcing patch-wise predictions in the second stage. Our predictions for Daraa, Deir-Ez-Zor, and Aleppo rely much more on repetition and temporal smoothing. We confirm the role of temporal smoothing in *SI Appendix*, Table S5 by varying temporal lags and providing performance estimates without spatial smoothing.

The improvements with temporal smoothing suggest that the domain-shift problem across time plays an important role when angles, lighting, vegetation, and seasons change. Our results therefore highlight the potential role of repeated high-frequency imagery and temporal smoothing for providing useful destruction monitoring. The extreme imbalance combined with small samples imposes serious trade-offs for monitoring, but we will show in the following section that monitoring can be brought to work even in the case of Aleppo, which has one of the more unbalanced samples in our dataset.

External Validation Exercises. We conduct two validation exercises to illustrate the merits of our approach. We first make use of the no-analysis areas in Aleppo (Fig. 1) that have been entirely excluded from the training process. One of these zones corresponds to the Ramouse neighborhood in the southernmost tip of our study area in Aleppo—an area that our classifier identified as heavily destroyed, as depicted in Fig. 3*C*.

In Fig. 4, we show satellite imagery from a subarea of the Ramouse neighborhood at two points in time, before (December 6, 2016; *A*–*C*) and after (December 18, 2016; *D*–*F*) a major heavy weaponry attack. We show raw satellite images (Fig. 4 *A* and *D*), patch-wise visualizations of the second-stage continuous prediction scores (Fig. 4 *B* and *E*), and a binary classification (Fig. 4 *C* and *F*). Due to the classifier not having been trained on this area, this exercise serves as a good out-of-sample validation test. Visual inspection of the raw images shows no destruction before (*A*), but extensive building destruction after the attacks (*D*). Comparing the continuous prediction scores before (*B*) and after the attack (*E*) shows a significant increase in predicted destruction by our approach, which coincides clearly with the locations of actual destruction of buildings in the area. Note that the model also classifies correctly areas without building destruction, such as the industrial compounds in the northeast and southwest of the image, as not destroyed at both points in time. The same applies to the fields and roads in the East and the forest in the West. Fig. 4 *A* and *D* shows one way of converting continuous prediction scores into a binary classification. The threshold chosen here is optimized to reach a level of 50% recall in the test sample. One can observe that the before period is consistently classified as nondestroyed (with one exception), whereas destruction is indicated in affected areas after the attacks.

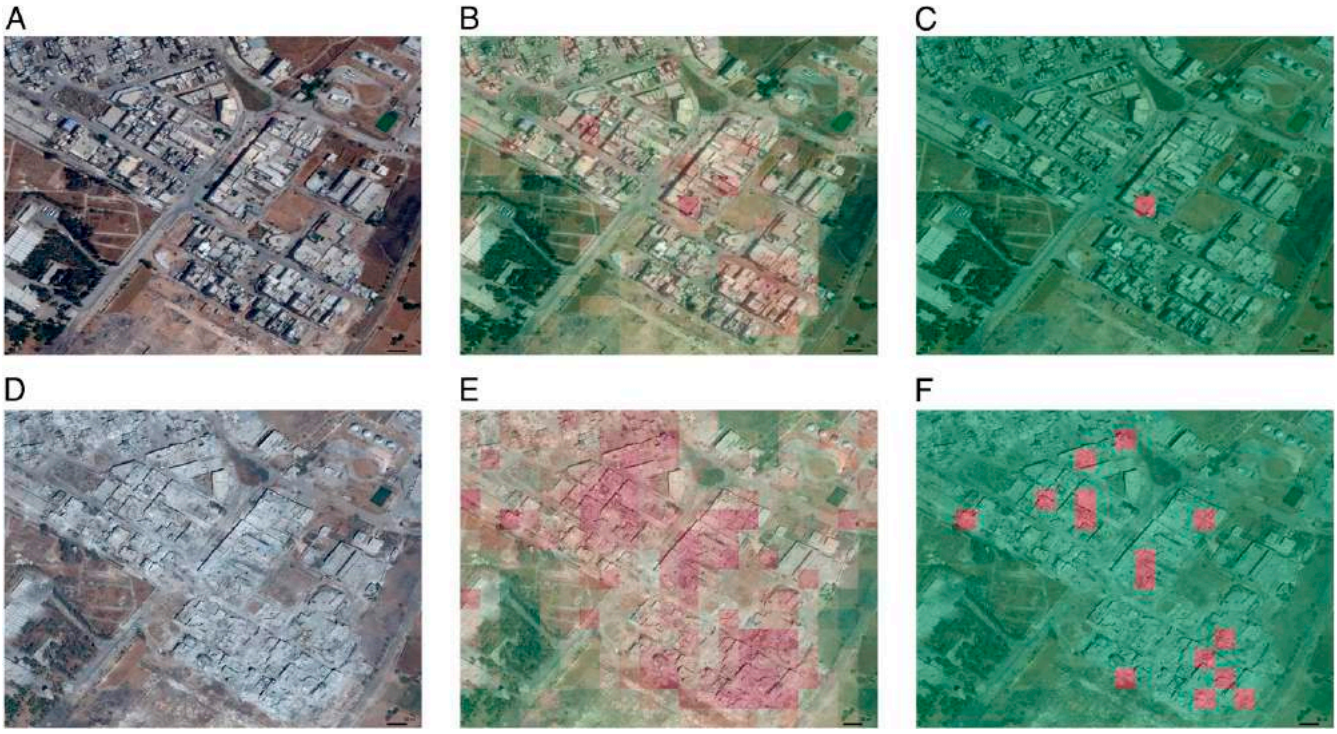


Fig. 4. Example of raw satellite images (*A* and *D*) and second-stage patch-wise continuous predictions scores (*B* and *E*) and binary classification (*C* and *F*) for the Ramouse neighborhood of Aleppo, Syria. Before (*A*–*C*) and after (*D*–*F*) heavy weaponry attacks are shown. Green color indicates low prediction scores, and red color indicates high prediction scores. Color bins reflect deciles of fitted values. The binary classification cutoff was optimized to reach 50% recall in the test sample. Satellite image recording dates: December 6, 2016 (before), and December 18, 2016 (after). Approximate image centroid location: 36.1525 decimal degrees north and 37.1332 east. Sources: Google Earth/Maxar satellite imagery and author calculations.

Fig. 4 demonstrates that the classifier is able to identify destruction in parts of the city that were not part of the training sample. This is important, as it shows that we are able to successfully solve the spatial and temporal domain-shift problems within Aleppo and, thus, generate a time series of destruction data in this way. If our automated method was to augment human monitoring, this is the kind of data that would be passed to human verification.

Given our strategy of expanding labels forward and backward in time, it becomes particularly important to verify the ability of our approach to approximate the timing of destruction. We therefore validated our dense predictions in an event-study framework, which relies on an external dataset of georeferenced bombing events in Syria. In particular, we relied on 731 bombing events with precise location information from the Live Universal Awareness Map project (LiveUAmap) (29). We merged these events with our pooled sample of dense predictions at the patch-time level. We then conducted an event-study regression on a sample of over 2.8 million observations to test whether our prediction scores increase in the aftermath of an externally reported bombing event (see *SI Appendix*, section 2 for details).

We present a coefficient summary plot for two second-stage modules in Fig. 5. The graph shows clearly that bombing events are positively and significantly correlated to the destruction scores at the time and patch levels. Note that the baseline hazard of destruction, i.e., the mean of the dependent variable, is very small in our sample (*SI Appendix*, Table S2). Compared to the baseline level of the respective destruction score, the point estimates imply an increase of 29% and 37%, respectively, after a bombing event is reported in a given cell. This is a substantial increase if one keeps in mind that not all bombing events will result in the destruction of a building, introducing attenuation bias in the regression. The figure also shows that temporal smoothing implies big gains in overall signal strength, with the coefficients from the full model represented by the red diamonds being consistently above the spatial only model as depicted by the blue squares.

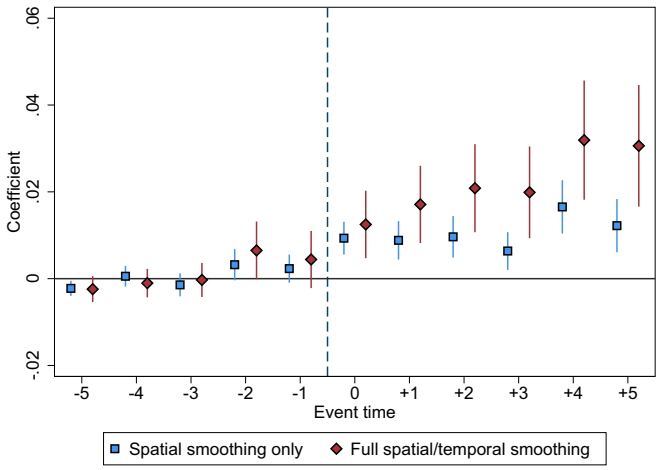


Fig. 5. Event-study calidation exercise pooled sample. External bomb-event data from LiveUAmap is positively and significantly correlated with satellite predicted war destruction at the patch level. The figure shows coefficients from a regression of five leads and lags of bombing events identified in the event data against our continuous destruction prediction score from the second stage. Point estimates depicted by blue squares correspond to second-stage continuous prediction scores with spatial smoothing only, and red diamonds correspond to the full model with spatial and temporal smoothing. Error bars represent 95% CIs. The dashed line indicates the occurrence of a bombing event in the event data, and coefficients capture the response in predicted damage. The full regression specification and results are reported in *SI Appendix*, section 2 and Table S2, respectively. Sources: LiveUAmap event data and author calculations.

Discussion

Building destruction due to heavy weapon attacks is a particularly salient form of war-related violence. Destruction is often used as a military strategy to displace population and is responsible for tremendous human suffering beyond the loss of life. Likewise, organizations like the Red Cross warn that massive destruction of urban infrastructure (also called *urbicide*) has dramatic knock-on effects on health, as it implies the destruction of water and power supplies, as well as hospitals. Therefore, reliable and updated data on destruction from war zones play an important role for humanitarian relief efforts, but also for human-rights monitoring, reconstruction initiatives, and media reporting, as well as the study of violent conflict in academic research. Studying this form of violence quantitatively, beyond specific case studies, is currently impossible due to the absence of systematic data.

Our method of identifying building destruction combines the existing state-of-the-art computer-vision methods with an additional postprocessing step and exploits the time dimension of destruction data to expand the training dataset. This allows us to exploit the repetition of imagery to bring down error rates when classifying destruction. Thanks to these advances, we were able to achieve an AUC of above 0.9 and an average precision of over 0.42 in the unbalanced sample from six Syrian cities. We also show that our approach is able to identify the timing and location of building destruction out-of-sample, i.e., in areas of Aleppo that have not been used for training the classifier.

These results are encouraging and allow applicability for automated destruction classification and even close to real-time tracking for policy purposes. Our method is particularly well-placed to take advantage of the ever-increasing temporal granularity of imagery. Our calculations suggest that human manual labeling of our entire dataset would cost approximately 200,000 USD, and additional repetitions of imagery would increase these costs almost proportionally. With an automated method like ours, higher image frequency helps precision and comes at only marginal extra costs. However, our results also suggest limitations where average precision falls, e.g., if only a very low share, less than 1%, of a city is destroyed. For applications requiring high precision in heavily imbalanced prediction problems such as the monitoring of several cities, we believe that the real use case for our approach will be in a decision-support framework, in which the predictions are combined with human verification to create much faster and accurate on-the-ground violence detection. Iterations between machine learning and human verification can also help in improving the training process (30) and could be easily integrated in our approach.

The performance of our method could be further improved by increasing the size of the training dataset, which could also help adapt it to classify destruction in other war zones around the globe. Further performance improvement could be achieved through *fine tuning*, a common practice in deep learning in which the network is pretrained with a large sample of building destruction from a variety of contexts in the first step and then refined by training on heavy weaponry destruction. This could be implemented by using a recent public dataset of natural-disaster destruction imagery that provides a sample of 98,000 annotated buildings across three levels of damage (31). Moreover, domain-adaptation techniques developed for deep learning could be used to try to further minimize the remaining domain biases (32).

Our label-augmentation technique is driven by strong assumptions and should therefore be regarded only as a first step in understanding the dynamic classification of building destruction over time. A particularly fruitful direction for future research could be to model the data-generating process of what we call the "uncertain class" between changing labels and after the last label date. This should then be combined with label smoothing to generate probabilistic labels (33). Such a holistic approach would also need to think about label priors regarding the reconstruction process. Future applications of such an approach would then be able to augment the human-classification process of verifying violence—so-called digital humanitarians (34)—and track the postwar recovery within the same classifier model.

The destruction data that can be generated with monitoring approaches such as the one presented in this article open up possibilities for a set of new research agendas in the social sciences (35). For example, our approach may advance the academic literature on understanding the microlevel determinants of violence (36–42). At what stage in a conflict is building destruction used? What can be done to reduce civilian fatalities during urban warfare? What are the effects of building destruction on displacement compared to other kinds of violence such as small firearms? Can reporting-based violence data be used to reduce error in the remote-sensing exercise, or can combined measures be developed (43, 44)? Can destruction data be used to reveal biases in reporting-based measures? An additional potential application of our method is conflict-forecasting systems, like the "Violence Early-Warning System," which rely on spatial violence dynamics in their forecasts (45).

Finally, there are important ethical concerns in war-destruction monitoring that should be considered. Research in the social sciences has shown that monitoring tends to reduce armed violence between states, but there are also examples where the opposite is true (46, 47). Theoretically, we can identify specific scenarios in which monitoring worsens the situation on the ground. If local actors are using the flow of information about atrocities to displace population and do not fear repercussions linked to the monitoring of these atrocities, then monitoring itself can increase violence and should, therefore, not be conducted publicly.

Materials and Methods

Analysis. The CNN model was built and trained in Tensorflow. Analysis was performed in QGIS, Python, R, and Stata.

Materials and Data Availability. Detailed explanations of methods in this study are provided in *SI Appendix*. All main code is available at GitHub (https://github.com/monitoring-war-destruction) (48). The repository provides all programming codes for image preprocessing and label augmentation, as well as first- and second-stage training and testing. All data is provided in the repository, except for the satellite imagery which cannot be provided due to copyright restrictions.

ACKNOWLEDGMENTS We thank Eli Berman, Joshua Blumenstock, Mathieu Couttenier, Joan Maria Esteban, Clément Gorin, Edward Miguel, Sebastian Schütte, and Jacob Shapiro for useful comments and discussions. We are grateful to Bruno Conte Leite, Jordi Llorens, Parsa Hassani, Dennis Hutschenreiter, Shima Nabiee, and Lavinia Piemontese for excellent research assistance. We are particularly grateful to Javier Mas for his research assistance, which produced the coding backbone to this project. We thank seminar participants at the Applied Machine Learning, Economics, and Data Science, University of California Berkeley, Empirical Studies of Conflict Project Annual Meeting, AI for Development conference by the Center for Effective Global Action and the World Bank Development Impact Evaluation Group University of Bozen/Bolzano, International Institute of Social Studies of Erasmus University, University of Economics Ho Chi Minh City, Lyon University, Trinity College Dublin, Barcelona Graduate School of Economics, Institute for Economic Analysis of the Spanish Council for Scientific Research, Universitat de Barcelona, Berlin Network of Labor Market Research Winter Workshop, PREVIEW workshop at the German foreign office, and Violence Early-Warning System workshop in Uppsala. A.G. and H.M. were supported by "La Caixa" Foundation Project Grant CG-2017-04, title: "Analysing Conflict from Space"; and by Spanish Ministry of Science and Innovation, through the Severo Ochoa Program for Centers of Excellence in R&D Grant CEX2019-000915-S. H.M. was supported by Spanish Ministry of Science, Innovation and Universities Grant PGC-096133-B-100. A.G. was also supported by Spanish Ministry of Science, Innovation and Universities Grant PGC2018-094364-B-100. J.H. and A.M. were supported by the Chapman University Faculty Opportunity Fund. A.M. was supported by the Smith Institute of Political Economy and Philosophy at Chapman University. Any remaining errors are our own.

1. F. Witmer, Remote sensing of violent conflict: Eyes from above. *Int. J. Rem. Sens.* **36**, 2326–2352 (2015).
2. L. Gueguen, R. Hamid, "Large-scale damage detection using satellite imagery" in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, New York, NY, 2015), pp. 1321–1328.
3. F. Kahraman, M. Imamoglu, and H. F. Ates, "Battle damage assessment based on self-similarity and contextual modeling of buildings in dense urban areas" in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)* (IEEE, New York, NY, 2016), pp. 5161–5164.
4. Y. LeCun, Y. Bengio, G. Hinton, Deep learning. *Nature* **521**, 436–444 (2015).
5. N. Jean *et al.*, Combining satellite imagery and machine learning to predict poverty. *Science* **353**, 790–794 (2016).
6. R. Engstrom, D. Newhouse, J. Hersh. Poverty from space: Using high resolution satellite imagery for estimating economic well-being and geographic targeting (World Bank Policy Research Working Paper 8284, World Bank, Washington, DC, 2017).
7. C. Yeh *et al.*, Using publicly available satellite imagery and deep learning to understand economic well-being in Africa. *Nat. Commun.* **11**, 1–11 (2020).
8. UNITAR Operational Satellite Applications Programme, Maps and Data. https://www.unitar.org/maps. Accessed 19 May 2021.
9. World Bank. The toll of war: The economic and social consequences of the conflict in Syria (Tech. Rep., World Bank, Washington, DC, 2017).
10. Amnesty International. Strike tracker. Decode how US-led bombing destroyed Raqqa, Syria (2020) https://decoders.amnesty.org/projects/strike-tracker. Accessed 19 May 2021.
11. N. P. Gleditsch, P. Wallensteen, M. Eriksson, M. Sollenberg, H. Strand, Armed conflict 1946-2001: A new dataset. *J. Peace Res.* **39**, 615–637 (2002).
12. C. Raleigh, A. Linke, H. Hegre, J. Karlsen, Introducing ACLED: An armed conflict location and event dataset: Special data feature. *J. Peace Res.* **47**, 651–660 (2010).
13. M. R. Sarkees, F. Wayman, *Resort to War: 1816-2007* (CQ Press, Washington, DC, 2010).
14. R. Sundberg, E. Melander, Introducing the UCDP georeferenced event dataset. *J. Peace Res.* **50**, 523–532 (2013).
15. M. Price, A. Gohdes, P. Ball, Documents of war: Understanding the Syrian conflict. *Significance* **12**, 14–19 (2015).
16. N. B. Weidmann, A closer look at reporting bias in conflict event data. *Am. J. Polit. Sci.* **60**, 206–218 (2016).
17. T. Pettersson, M. Öberg, Organized violence, 1989–2019. *J. Peace Res.* **57**, 597–613 (2020).
18. K. He, X. Zhang, S. Ren, J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification" in *2015 IEEE International Conference on Computer Vision (ICCV)* (IEEE, New York, NY, 2015), pp. 1026–1034.
19. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition. arXiv [Preprint] (2015). https://arxiv.org/abs/1409.1556 (Accessed 19 May 2021).
20. A. J. Cooner, Y. Shao, J. B. Campbell, Detection of urban damage using remote sensing and machine learning algorithms: Revisiting the 2010 Haiti earthquake. *Rem. Sens.* **8**, 868 (2016).
21. L. Gueguen, R. Hamid, Toward a generalizable image representation for large-scale change detection: Application to generic damage analysis. *IEEE Trans. Geosci. Rem. Sens.* **54**, 3378–3387 (2016).
22. F. Kahraman, M. Imamoglu, H. F. Ates, Disaster damage assessment of buildings using adaptive self-similarity descriptor. *Geosci. Rem. Sens. Lett. IEEE* **13**, 1188–1192 (2016).
23. J. Yuan, Automatic building extraction in aerial scenes using convolutional networks. arXiv [Preprint] (2016). https://arxiv.org/abs/1602.06564 (Accessed 19 May 2021).
24. N. Attari, F. Ofli, M. Awad, J. Lucas, S. Chawla, "Nazr-CNN: Fine-grained classification of UAV imagery for damage assessment" in *2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)* (IEEE, New York, NY, 2017), pp. 50–59.
25. A. Fujita *et al.*, "Damage detection from aerial images via convolutional neural networks" in *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)* (IEEE, New York, NY, 2017), pp. 5–8.
26. F. Nex, D. Duarte, F. G. Tonolo, N. Kerle, Building damage detection with deep learning: Assessment of a state-of-the-art CNN in operational conditions. *Rem. Sens.* **11**, 2765 (2019).
27. B. Sun, K. Saenko, "Deep CORAL: Correlation alignment for deep domain adaptation" in *ECCV 2016 Workshops: European Conference on Computer Vision*, G. Hua, H. Jégou, Eds. (Lecture Notes in Computer Science, Springer, Cham, Switzerland, 2016), vol. 9915, pp. 443–450.
28. *Google Earth*. https://www.google.com/earth, Accessed 7 September 2020.
29. Live Universal Awareness Map Syria. https://syria.liveuamap.com/. Accessed 19 May 2021.
30. M. Colaresi, Z. Mahmood, Do the robot: Lessons from machine learning to improve conflict forecasting. *J. Peace Res.* **54**, 193 (2017).
31. R. Gupta *et al.*, "Creating xBD: A dataset for assessing building damage from satellite imagery" *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. https://openaccess.thecvf.com/content_CVPRW_2019/papers/cv4gc/Gupta_Creating_xBD_A_Dataset_for_Assessing_Building_Damage_from_Satellite_CVPRW_2019_paper.pdf. Accessed 19 May 2021.
32. G. Csurka, "Domain adaptation for visual applications: A comprehensive survey" in *Advances in Computer Vision and Pattern Recognition*, G. Csurka, Ed. (Springer, Cham, Switzerland, 2017), pp. 1–35.
33. R. Müller, S. Kornblith, G. Hinton, When does label smoothing help? arXiv [Preprint] (2020). https://arxiv.org/abs/1906.02629 (Accessed 19 May 2021).
34. P. Meier, *Digital Humanitarians: How Big Data Is Changing the Face of Humanitarian Response* (Routledge, London, UK, 2015).
35. K. S. Gleditsch, N. W. Metternich, A. Ruggeri, Data and progress in peace and conflict research. *J. Peace Res.* **51**, 301–314 (2014).
36. T. Besley, H. Mueller, Estimating the peace dividend: The impact of violence on house prices in Northern Ireland. *Am. Econ. Rev.* **102**, 810–833 (2012).
37. O. Dube, J. F. Vargas, Commodity price shocks and civil conflict: Evidence from Colombia. *Rev. Econ. Stud.* **80**, 1384–1421 (2013).
38. M. Burke, S. M. Hsiang, E. Miguel, Climate and conflict. *Annu. Rev. Econ.* **7**, 577–617 (2015).
39. S. Michalopoulos, E. Papaioannou, The long-run effects of the scramble for Africa. *Am. Econ. Rev.* **106**, 1802–1848 (2016).
40. N. Novta, Ethnic diversity and the spread of civil war. *J. Eur. Econ. Assoc.* **14**, 1074–1100 (2016).
41. N. Berman, M. Couttenier, D. Rohner, M. Thoenig, This mine is mine! How minerals fuel conflicts in Africa. *Am. Econ. Rev.* **107**, 1564–1610 (2017).
42. M. Manacorda, A. Tesei, Liberation technology: Mobile phones and political mobilization in Africa. *Econometrica* **88**, 533–567 (2020).
43. J. V. Henderson, A. Storeygard, D. N. Weil, Measuring economic growth from outer space. *Am. Econ. Rev.* **102**, 994–1028, (2012).
44. K. Lum, M. E. Price, D. Banks, Applications of multiple systems estimation in human rights research. *Am. Statistician* **67**, 191–200 (2013).
45. H. Hegre *et al.*, ViEWS: A political violence early-warning system. *J. Peace Res.* **56**, 155–174 (2019).
46. G. Gordon, Violence and intervention. PhD thesis, Columbia University, New York, NY (2016).
47. B. R. Early, E. Gartzke, Spying from space: Reconnaissance satellites and interstate disputes. *J. Conflict Resolut.*, https://doi.org/10.1177/0022002721995894 (2021).
48. H. Mueller, A. Groeger, J. Hersh, A. Matranga, J. Serrat, Monitoring war destruction. GitHub. https://github.com/monitoring-war-destruction. Deposited 19 May 2021.

PNAS
www.pnas.org

Supplementary Materials for

Monitoring War Destruction from Space Using Machine Learning

Hannes Mueller, Andre Groeger, Jonathan Hersh, Andrea Matranga, and Joan Serrat

Correspondence to: andre.groger@uab.es or hannes.mueller@iae.csic.es

This PDF file includes:

Supplementary text
Figs. S1 to S5
Tables S1 to S5
SI References

Supporting Information Text

1. Materials and Methods

A. Methodology. We use a flexible CNN architecture, which is summarized graphically in Figure S3. Given this structure, we optimize over a set of hyper-parameters to determine the optimal network architecture. Each network uses a series of convolutional blocks. We use convolution because objects in an image correspond to a particular spatial arrangement of pixels and modeling images therefore requires a method that captures the local interactions between input pixels. The idea of convolution is to set neighboring pixels of a location in a relationship to each other through filters which stride over the image and which provide local summaries of the image.

Each block contains: (i) a convolutional layer, followed by (ii) a rectified linear unit (ReLu) activation function, (iii) a max-pooling layer and (iv) a final dropout layer. The output of the last convolutional block is flattened to a vector and input to two successive fully connected layers followed by a non-linear activation, with a batch normalization layer between them. We explore the space of parameters formed by the number of convolutional blocks, the size of the convolutional kernels, the *stride* of max-pooling, the probability of dropout, the number of units in the fully connected layers, and the type of non-linear activation (which can be *ReLU* or *sigmoid*).

Instead of fixing the network architecture and learn the convolutional filters we have also optimized over the architecture itself. In addition to the usual hyper-parameters like the number of epochs and the learning rate, we consider several architectural parameters. The optimal hyper-parameters for our architecture are reported in Table S1.

The final step of the first-stage is to apply the trained model (trained on the training sample) to produce predictions for every patch in every post image as in practical applications of monitoring destruction. In this step we evaluate the full sample of over 3 million image patches (see Table 1) and produce a continuous prediction score for all patches and times in all cities we analyze. The result is what we call *dense* predictions and this forms the raw material for the second stage of our methodology. However, when reporting on performance we always rely on training on only 70% of the patches in the training sample and report performance in the 30% test sample.

B. Description of Second Stage. In the second stage, to exploit the temporal and spatial clustering of destruction, we train a random forest model including different combinations of spatial and time leads and lags for each patch to improve precision. The logic behind this is that destruction is not only serially correlated, but also spatially clustered. Indeed, it can be shown in the dense prediction data that both the spatial environment and time lags are extremely informative for whether a patch has been destroyed. We separate this step from the deep learning stage for maximum flexibility and modularity. We use a random forest model to exploit both information on the mean prediction scores around an image and their standard deviation. In the results section we show how this step improves precision. To generate a binary prediction from the random forest model we pick a cutoff that reaches 50 percent recall in the training sample.

We proceed as follows. Based on the raw continuous prediction scores from the CNN first stage, we calculate several spatial features: the average number of predicted values and their standard deviation for one spatial lag (the four closest patches) and two spatial lags (the twelve closest patches). We also use the number of patches surrounding the patch to control for observations at the edges of the image. This is what we call spatial smoothing. In addition we use the continuous predicted values of two temporal forward lags and two temporal backward lags for the same patch. We call this temporal smoothing.

At any moment of the second stage we maintain the train/test split when training the random forest model. The features are used in a random forest model with 300 trees, max_depth=15, min_samples_split=2, min_samples_leaf=20 to produce the outcome of the second stage. The trained model is evaluated using the test sample. After producing the test statistics we re-train the model, maintaining the hyperparameters but now using the entire dense prediction sample. This is the final output that we use for the validation exercises and it covers the entire sample of over 3 million patches.

2. Validation Exercise

To externally validate our measure of destruction, we construct an event study relying on bombing event data with precise geolocation from the Live Universal Awareness Map project (LiveUAmap). This data is available for the whole of Syria starting in 2016. For our sample of Syrian cities the database reports 731 bombing events during the years 2016 and 2017. We download and merge these events with our pooled city sample (see Table 1) at the patch-time level. Due to the widely different number and frequency of temporal observations available for each city, we merge the sample at the first time period available in 2016 for each city. We replace missing values by zeros for all observations up until 5 event periods before the first observation in 2016 and 5 periods after the last observation in 2017. We then construct a time-varying binary indicator of bombing events at the cell level as well as 5 leads and lags of this indicator. Based on the availability of data and the construction of the dataset outlined, we are left with a balanced sample of around 2.8 million observations in this exercise.

We run the following regression specification:

$$destruction_{it} = \sum_{j=-5}^{+5} \beta_j bomb_{it}^{j} + post_{it} + \mu_i + \theta_{ct} + \epsilon_{it}$$

, with *destruction* being the continuous destruction prediction score from the second stage for each cell i and date t. *bomb* is the binary indicator for a bombing event reported in the LiveUAmap data occurring in a specific cell during period $j \in [-5, +5]$ periods away from t. $post_{it}$ is a dummy being equal to one for treated cells in time periods after the fifth lag. In this setup, the omitted categories are those preceeding the fifth lead and, consequently, the coefficients are normalized at the average destruction level during the pre-periods. μ_i and θ_{ct} are cell and city-time fixed effects. Robust standard error are clustered at the cell level.

Table S2 reports the regression results from this exercise as depicted in Figure 5 of the article. The dependent variable in specification (1) is the second-stage destruction score using only spatial smoothing, whereas in specification (2) it is the full second-stage model relying on spatial and two temporal leads and lags. The coefficient estimates show clearly that the LiveUAmap external bombing events are positively and significantly correlated with our second-stage destruction scores when and where bombing events are reported in a specific cell. Comparing the point estimates between the two columns shows that the absolute magnitudes of the coefficients are, on average, larger in column (2) which is in line with a higher signal based on the superior model performance.

Note that the baseline hazard of destruction is very small as reflected by the mean of the dependent variable reported in the last line. Compared to the baseline level of the respective destruction score, the point estimates in specification (1) and (2) imply an increase of 29% and 37%, respectively, when a bombing event is reported in a given cell, which is significant both in qualitative as well as quantitative terms. An important thing to add here is that not all bombing events lead to the destruction of a building, which can be picked up by our classier. The coefficient is, therefore, a mix of destruction and no destruction

results of reported bombing events. See, for example, Figure S1 below to get an impression of visual destruction footprints in images that were annotated as destructed by UNOSAT/UNITAR.

3. Glossary

Area Under the Curve (AUC). A receiver operating characteristic curve, or ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier model. The curve combines four statistics: false positives fp (true negatives classified as positives), false negatives fn (true positives classified as negatives), true positives tp and true negatives tn. These four statistics can be calculated at each threshold for a positive classification. With a higher threshold the number of positive predictions falls and the number of negative predictions increases. The ROC curve shows the trade-off between the true positive rate (or recall) given by $tp/(tp+fn)$ on the y-axis and the false positive rate given by $fp/(fp+tn)$ on the x-axis at every threshold. (See (1)). The AUC is the area under this curve and therefore represents a performance measure agnostic to the specific value of the threshold chosen.

Class imbalance. In a supervised classification problem, class imbalance occurs when one or more classes have much less samples than the other classes. In other words, the distribution of samples among classes is highly uneven. This poses a challenge to many classification techniques because, implicitly, they assume a more or less even distribution. Most chances are the classifier will learn to ignore the minority classes because they may be a tiny fraction of the training set and prediction errors do not contribute much to the cost function in optimization. These kind of problems also preclude the direct use of standard accuracy metrics.

Convolutional neural network (CNN). CNNs are a specialized kind of neural network for processing data that has a known grid-like topology. Examples include time-series data, which can be thought of as a 1-D grid taking samples at regular time intervals, and image data, which can be thought of as a 2-D grid of pixels (See (2), chapter 9). A CNN has one or more layers performing a convolution, which is, loosely speaking, a kind of weighted average. Quite often convolutional layers are followed by non-linear layers like ReLu or sigmoid, and max pooling.

Cross-validation. In the context of supervised classification with a low number of samples, k-fold cross-validation is a strategy to avoid overfitting to the training set. The training set is split into k smaller sets, and k "folds" are made, each one taking one of the subsets as validation set and the remaining $k-1$ subsets as the training set. Then, one trains and assess the classifier with each fold, and computes the performance as an average of performance values for each learned classifier on its validation fold (See (1)).

Data augmentation or dataset augmentation is a simple technique to increase the size and variability of the training set, with the aim of improving the generalization of a predictor. In the case of computer vision where samples are images, data augmentation means to randomly (*i.e.* with a certain probability) perform transformations to the images in the training set like translation, rotation, horizontal and vertical flipping, color modification, cropping, adding noise, and even randomly erasing some parts, thus generating new samples.

Batch-normalization is a common type of layer in deep neural networks. It reparametrizes (re-centers and re-scales) its input so that it has zero mean and unit variance along certain dimensions. It is a complex transformation and its description and effect is beyond the scope of this short glossary. We refer the reader to ((2), chapter 8) for a complete exposition.

Domain adaptation. A domain shift occurs when a classifier is trained with a dataset following a certain probability distribution and then is applied to a test set, or samples at deployment time, with a different but related probability distribution. Many factors may cause a domain shift. In computer vision applications, examples include images acquired by a different camera/sensor, with different lighting conditions, or at a different environment. The classifier may then perform poorly because it can not generalize to the new domain. Domain adaptation techniques can be applied to prevent this problem to some extent. For a comprehensive review of this subfield of computer vision, we refer the reader to (3).

Dropout. A dropout layer simply ignores a fixed fraction $p \in (0, 1)$ of the input units and leaves intact the rest during training. At inference time, however, all units are considered, as if the dropout layer was not present. It is a simple yet powerful technique in deep learning intended to prevent overfitting and increase the generalization capability of a neural network.

False positive rate. The false positive rate is the share of 0s in the data that the classifier wrongly flagged up as positives or 1s. Formally, it is given by the ratio of false positives fp (true negatives classified as positives) relative to true negatives tn (true negatives that were classified as negatives) - $fp/(fp + tn)$.

Fine tuning. Training a large neural network from scratch, with the weights initialized from random values, is normally difficult and time consuming. One strategy to proceed faster that works well in practice is fine tuning. First, take some pre-trained network, that is, its architecture and learned coefficients on some different dataset. Then adapt it to a specific task, for instance if it is a classifier, replace the last fully-connected layer by one with the right number of outputs (classes). Then, start training the network with your dataset: the new last layer will be trained from scratch but previous layers, initialized at the previously learned weights, will be fine tuned. It often turns out that they are a good initial parameters for the new classification task. Fine tuning is a type of "transfer learning", from a source to a target dataset or task.

Fully-connected layer in its simplest version is the linear transform $y = Wx$ from an input $x = (x_1, x_2 \dots x_m)$ to a vector output $y = (y_1, y_2 \dots y_n)$, being W an $n \times m$ matrix of weights. With non-linear activation functions, they are the basic component of feed-forward neural networks.

Hyper-parameters. Most deep learning architectures and training algorithms come with parameters that one has to tune to the current task. For instance, if one includes dropout layers we have to provide a probability value. For example, an important hyperparameter is the learning rate. Selecting good starting values for the hyperparameters is key, and there are manual and automatic methods for it see ((2), chapter 11).

Maxpool. A typical block in a convolutional neural network is the succession of three layers: a convolutional one, a non-linear activation function (like ReLU) and a pooling layer. The latter replaces the output of the previous layer with a summary statistic of the nearby outputs. The max pooling operation reports the maximum output within a rectangular neighborhood (2). In this context the stride is the step in the spatial dimensions where the operation is performed, that is, the maximum within the rectangular neighborbood is computed for a grid of coordinates with separation equal to the stride.

Precision-recall curve. Precision-recall curves are alternatives plots to ROC curves when showing summaries of false positives fp (true negatives classified as positives), false negatives fn (true positives classified as negatives), true positives

tp and true negatives tn. On the y-axis the curve shows precision given by $tp/(tp + fp)$ and on the x-axis it shows the true-positive rate or recall given by $tp/(tp + fn)$. Intuitively, precision is the ability of the classifier not to label as positive a sample that is negative, and recall is the ability of the classifier to find all the positive samples (See (1)).

Typically, there is an inverse relationship between precision and recall: it is possible to increase one at the cost of reducing the other by changing some parameter of the classifier. One can show to which extent this occurs through the precision-recall curve, a curve formed by pairs of (precision, recall) points obtained from the classifier by varying this parameter.

Random forest classifier. A random forest is a type of predictor that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. It is a type of ensemble classifier, meaning that a diverse set of classifiers is created by introducing randomness in the classifier construction. The prediction of the ensemble is given as the averaged prediction of the individual classifiers (1).

ReLU (for rectified linear unit), is the non-linear activation function $f(x) = \max(0, x)$, typically inserted between successive linear layers, like feed-forward or convolutional (2).

Semantic segmentation. Given a pre-defined list of object classes, the semantic segmentation of an image is the process of assigning to each pixel the label of the object class it belongs to. Thus, it's a type of classification where the entities for which labels are predicted are the image pixels. All pixels assigned to a certain class thus receive the same label, even though they belong to two different objects (also called instances) of that class. For a recent review of modern techniques see (4).

Sigmoid. The logistic sigmoid $\sigma(x) = \dfrac{1}{1 + \exp(x)}$ is, like ReLU (see above) a non-linear activation function. It rescales the input to the range $(0, 1)$.

Supervised classifier. A supervised classifier is a type of machine learning algorithm that maps samples to a set of class labels. The parameters of the mapping are learned from examples, also called samples. Samples are object data. e.g. images or image features, usually adopting the shape of a vector. The learning algorithm that finds the classifier parameters takes as input training samples, that is, a list of pairs (sample data, class label).

True positive rate. The true positive rate is the share of 1s in the data that the classifier correctly identified as positives or 1s. Formally, it is given by the ratio of true positives tp (true positives classified as positives) relative to false negatives fn (true positives that were classified as negatives) - $tp/(tp + fn)$.

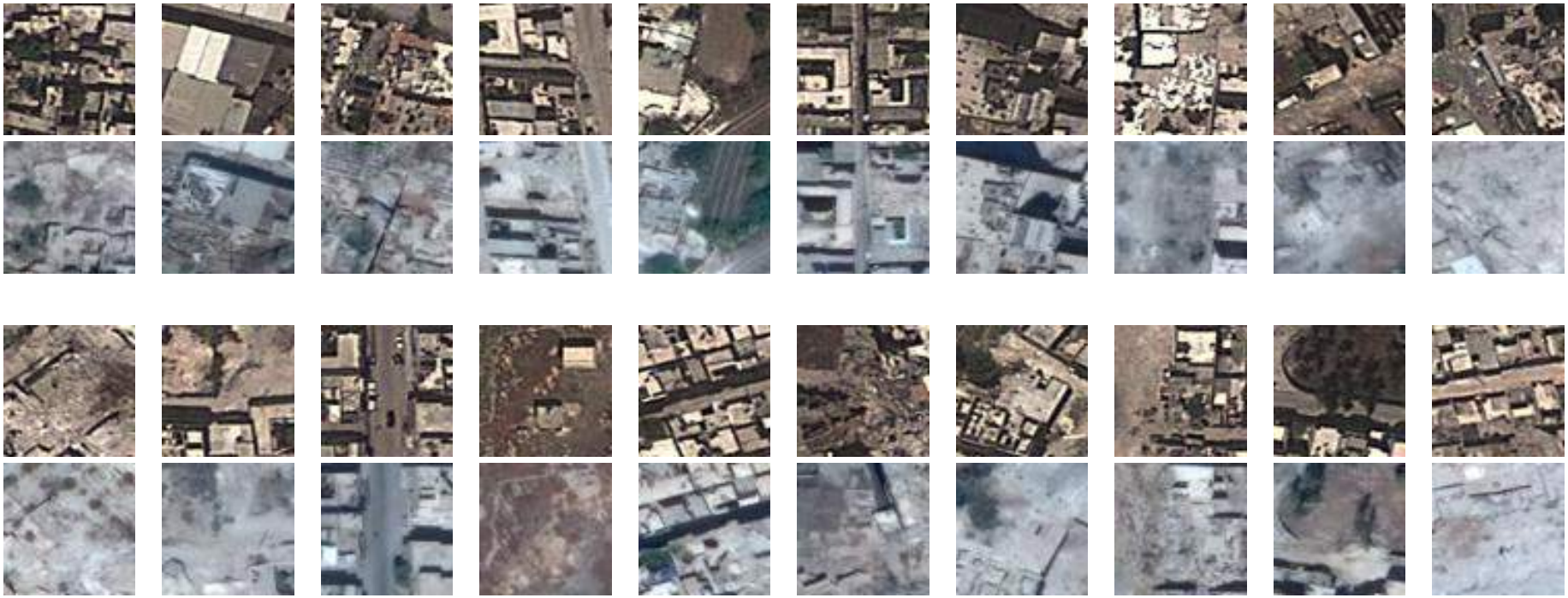


Fig. S1. Sample Imagery Pacthes Containing Destroyed Annotations, Pre- (2013) and Post- (2016) Syrian Civil War. Pre- samples are shown in odd rows, post- examples are shown in even rows. Images are of dimension 64×64 pixels, each containing at least one point annotated as *destroyed* in the UNITAR/UNOSAT damage assessment for Aleppo. Sources: Google Earth/Maxar Technologies.

Fig. S2. Sample Imagery Tiles Containing Non-Destroyed Annotations, Pre- (2013) and Post- (2016) Syrian Civil War. Pre- samples are shown in odd rows, post- examples are shown in even rows. Negatives samples indicate UNITAR/UNOSAT labels are not in *Destroyed* class. Bottom columns 1-5 show imagery tiles labeled with *Moderate Damage* and columns 6-10 show tiles containing *Severe Damage*, both of which are categorized in our target labels as non-destroyed. Sources: Google Earth/Maxar Technologies.

Fig. S3. Network architecture of CNN deep learning model. The ultimate architecture of the network is determined through sampling randomly a set of parameters and assessing its performance on the training set. See text and Table S1 for a description of parameters n, k_l, f_l, s_l, p_l, u and activation.

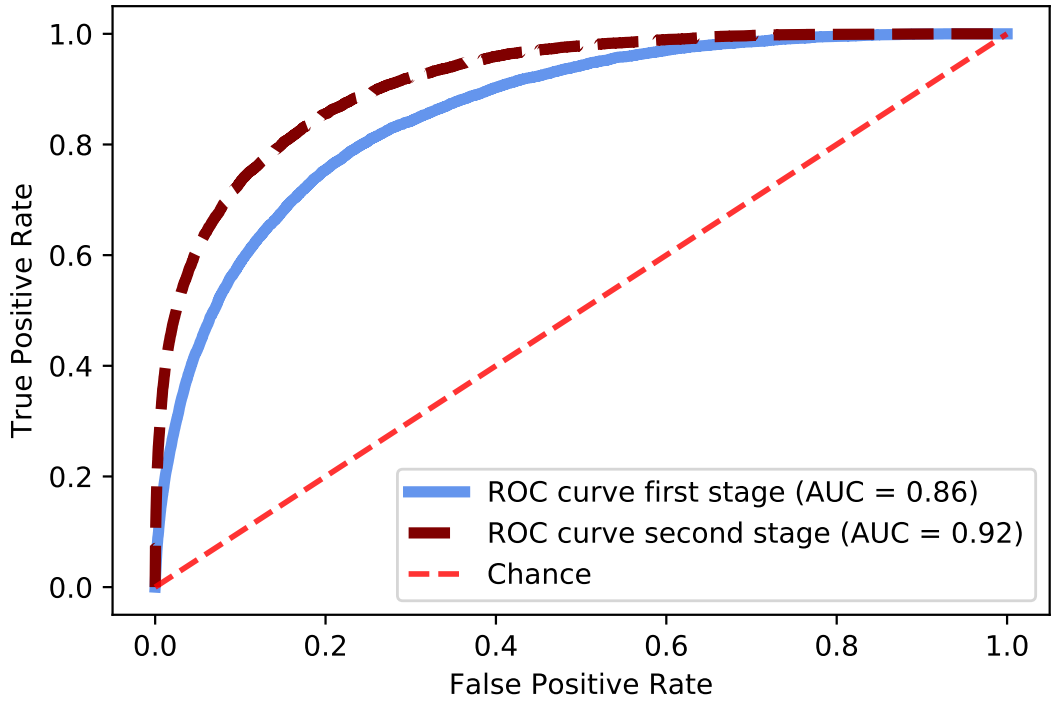
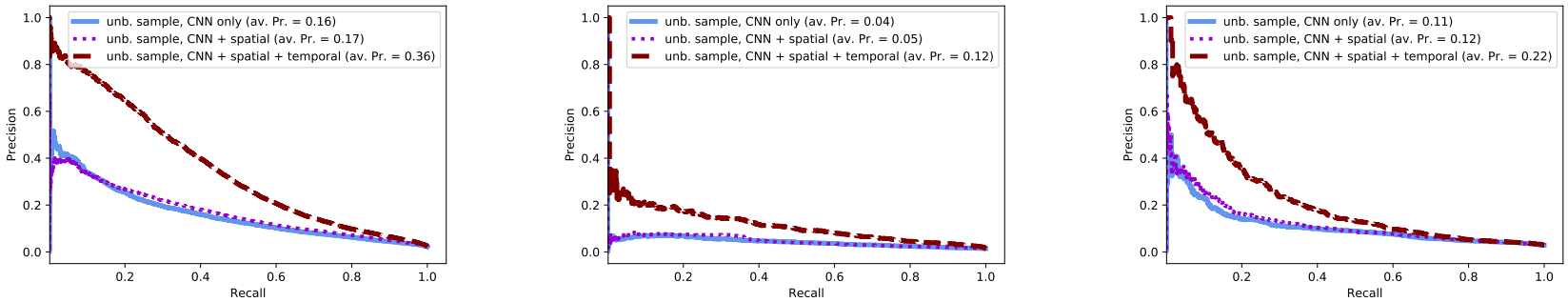


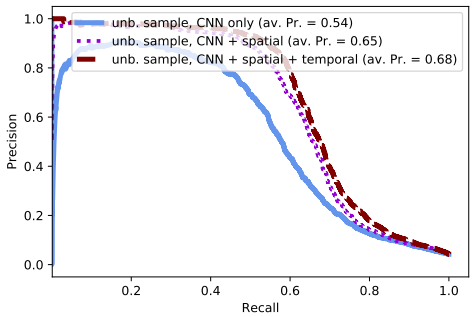
Fig. S4. First and second stage ROC curves for Aleppo: the displayed performance is in 30 percent test sample. Training and test sample are determined at the patch level so that the same patch is not used for training and testing. Sources: Author calculations.



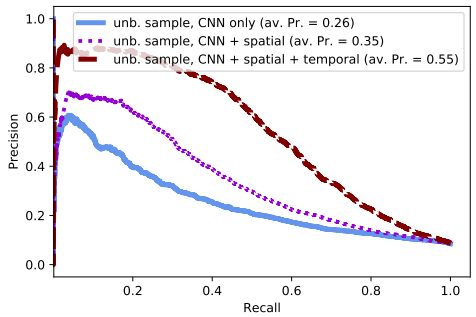
(a) Aleppo

(b) Daraa

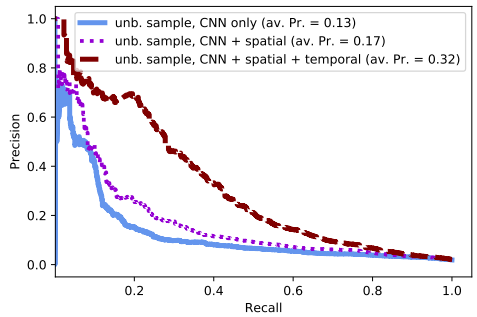
(c) Deir-Ez-Zor



(d) Hama



(e) Homs



(f) Raqqa

Fig. S5. Performance across six Syrian cities: Precision-recall curves for six Syrian cities. Performance is in the test sub-set of patches belonging to the respective city. Training is always with 70 percent of samples from all cities. Sources: Author calculations.

Parameter	Description	Value
n	number of convolutional blocks	2
$k_l, f_l\ l = 1 \dots n$	spatial size and the number of kernels of the convolutional layer in each block	$k_1 = 9, k_2 = 9, f_1 = 128, f_2 = 256$
s_l	stride of maxpooling	$s_1 = s_2 = 8$
p_l	probability of dropout	$p_1 = p_2 = 0.255$
activation	activation functions after the fully connected layers, either ReLu or sigmoid	ReLu

Table S1. Network parameters

Table S2. Event Study Validation Exercise Pooled Sample. External bomb event data from Liveuamap is positively and significantly correlated with satellite predicted destruction at the cell level.

VARIABLES	(1)	(2)
	Second-stage prediction scores	
	spatial	spatial & temporal
LiveUAmap bombing event in 5 waves	-0.00224**	-0.00242
	(0.000886)	(0.00151)
LiveUAmap bombing event in 4 waves	-0.000558	-0.00103
	(0.00122)	(0.00167)
LiveUAmap bombing event in 3 waves	-0.00143	-0.000291
	(0.00135)	(0.00200)
LiveUAmap bombing event in 2 waves	00321*	0.00651*
	(0.00183)	(0.00339)
LiveUAmap bombing event in 1 wave	0.00231	0.00442
	(0.00166)	(0.00336)
LiveUAmap bombing event	0.00933***	0.0125***
	(0.00193)	(0.00397)
LiveUAmap bombing event lag 1 wave	0.00884***	0.0171***
	(0.00226)	(0.00454)
LiveUAmap bombing event lag 2 waves	0.00964***	0.0208***
	(0.00243)	(0.00518)
LiveUAmap bombing event lag 3 waves	0.00637***	0.0199***
	(0.00221)	(0.00539)
LiveUAmap bombing event lag 4 waves	0.0165***	0.0319***
	(0.00313)	(0.00700)
LiveUAmap bombing event lag 5 waves	0.0122***	0.0306***
	(0.00312)	(0.00715)
LiveUAmap bombing event all posterior lags	-0.0404***	-0.104***
	(0.00405)	(0.00981)
Constant	0.0321***	0.0467***
	(2.87e-06)	(6.40e-06)
Patch/cell fixed effect	YES	YES
City-time/image fixed effect	YES	YES
Patches/cells	219,609	219,609
Observations	2,865,686	2,865,686
R-squared	0.684	0.705
Mean dependent variable	0.0321	0.0467

Sources: Author calculations based on second stage continuous destruction predictions scores and georeferenced LiveUAamap bombing event data. Dependent variable in column (1) is second stage prediction score with spatial smoothing only. Dependent variable in column (2) is second stage prediction score with spatial and temporal smoothing including 2 leads and lags. Note: Coefficients reported in each column come from a single fixed effect regression. Robust standard errors in parentheses, clustered at the cell level. *** $p<0.01$, ** $p<0.05$, * $p<0.1$

Table S3. Results across Syrian cities using only Aleppo labels to train model

City	(1) First-stage (CNN)	(2) Second-stage (CNN+RF)	(3) Second-stage (CNN+RF)	(4) Second-stage (CNN+RF)
	raw	with spatial leads/lags	with spatial & temporal leads/lags	with spatial & temporal leads/lags
	precision	*precision*	*precision*	*AUC*
Aleppo	11.3	12.5	33.3	89.6
Daraa	8.4	8.8	13.6	84.5
Deir-Ez-Zor	4.8	5.0	6.5	65.5
Hama	22.3	21.6	32.5	79.5
Homs	13.3	14.2	18.3	63.6
Raqqa	3.0	3.0	2.7	58.1

Note: first-stage predictions from convolution neural network (CNN) and second-stage predictions from random-forest model (CNN+RF) with spatial leads/lags (column 2) and spatial and two temporal leads/lags (columns 3 and 4). Columns (1) through (3) report the average precision and column (4) the "Area Under the Curve" (AUC). Aleppo performance reflects test samples withheld from training. Sources: Author calculations based on Google Earth/Maxar satellite imagery and UNITAR/UNOSAT damage annotations.

Table S4. UNITAR/UNOSAT Ground Truth Annotations For Buildings With Any Level of Destruction Over Time

City	(1) Buildings with any level of destruction followed by no change in level of destruction		(2) Buildings without any level of destruction followed by any level of destruction		(3) Buildings with any level of destruction followed by an increase in severity of destruction		(4) Buildings with any level of destruction that experienced a decrease in severity of destruction	
Aleppo	17,303	35.4%	30,554	62.4%	619	1.3%	461	0.9%
Daraa	1,598	66.8%	763	31.9%	14	0.6%	18	0.8%
Homs	8,451	67.1%	3,690	29.3%	209	1.7%	246	2.0%
Hama	9,568	85.8%	1,493	13.4%	9	0.1%	88	0.8%
Deir ez zor	8,018	70.1%	2,822	24.7%	244	2.1%	355	3.1%
Raqqa	708	33.0%	1,327	61.8%	12	0.6%	100	4.7%
Total	45,646	51.5%	40,649	45.8%	1,107	1.2%	1,268	1.4%

Sources: Number and percentage of buildings that suffered no change in damage level, an increase, decrease or are newly damaged, between two successive classifications by human coders according to UNOSAT/UNITAR ground-truth data. Percentages are compared to all observations that ever are damaged/destroyed.

Table S5. Detailed precision performance when varying second-stage module in the unbalanced sample

City	(1) First-stage (CNN) raw	(2) Second-stage (CNN+RF) with spatial lag only	(3) Second-stage (CNN+RF) with 2 temporal leads/lags only	(4) Second-stage (CNN+RF) with spatial & 1 temporal lead/lag	(5) Second-stage (CNN+RF) with spatial & 2 temporal leads/lags
Aleppo	16.1	16.9	33.2	28.3	35.7
Daraa	4.2	4.6	9.9	9.4	11.7
Deir-Ez-Zor	11.0	12.1	19.9	18.4	21.7
Hama	54.5	65.2	64.7	67.5	68.0
Homs	25.8	34.9	54.4	44.3	55.2
Raqqa	12.8	17.4	27.4	21.4	32.1
All	24.5	28.7	40.4	37.3	42.5

Sources: Author calculations based on Google Earth/Maxar satellite imagery and UNITAR/UNOSAT damage annotations. Note: Precision statistics for first-stage predictions from convolution neural network (CNN) and second-stage predictions from random-forest model (CNN+RF) with spatial leads/lags (column 2), two temporal leads/lags (column 3), spatial and one temporal lead/lag (column 4), and spatial and two temporal leads/lags (column 5).

References

1. Scikit-learn User Guide (https://scikit-learn.org/stable/user_guide.html) (2021) Accessed: January 2021.
2. I Goodfellow, Y Bengio, A Courville, *Deep Learning*. (MIT Press), (2016) http://www.deeplearningbook.org.
3. G Csurka, *A Comprehensive Survey on Domain Adaptation for Visual Applications*. (Springer International Publishing), (2017).
4. S Minaee, et al., Image segmentation using deep learning: A survey (2020) arXiv:2001.05566 (15 November 2020).