Satellite Image Analysis for Building Detection and Natural Disaster Damage Classification

Cade Crow, Melvin Juwono, Abuzar Royesh, Tsion (T) Tesfaye
Stanford University
ccrow00 | mjuwono | aroyesh | ttesfaye

Abstract

In the aftermath of natural disasters, timely detection of where damages have happened and dispatching rescue teams can save thousands of lives. To determine where efforts need to be focused, a human assessment of damage is usually required. These assessments take time and money and have inherent risk. To try and solve this problem, we worked on a two-step model that would take satellite images as input, detect buildings, and determine whether or not a building is critically damaged to warrant attention. Utilizing a ResNet50 infrastructure, we were able to obtain high training accuracy on damage classification, above what we expect a human can achieve. However, the accuracy on our testing set showed that improvements can be made to reduce variance. Our attempt at using L2 weights regularization is promising and given the right parameter value it is expected that the model will not overfit to the training data and perform better on the testing data. For building detection, we use Mask R-CNN (based on FPN for object detection and RESNET101 backbone). However, our model's performance was suboptimal in part due to computational challenges and time restrictions we faced.

1 Introduction

We aim to build a model that can be deployed to assist natural disaster relief efforts in developing countries. Our vision is that governments and non-governmental organizations can use satellite image data to assess which cities, villages, and infrastructure is in most critical need of support following a natural disaster. We hope that the use of a deep learning model would eliminate the danger, time, and expenses associated with in-person assessments so that relief could be quickly dispatched where it is most needed.

We see our task as having 3 main steps in the overall process. The first step is building detection. The input to our first step's algorithm is a satellite image, which is processed using a mask R-CNN and outputs x, y coordinates outlining the buildings in the image. For our second step, we perform image manipulation to crop images to a standard size and isolate buildings before feeding them into our second model. We use the x, y coordinate outputs from the first model to identify the locations to crop and isolate buildings in each image. For our final step, we use the cropped images as input to our ResNet model, which predicts the damage sustained by each building and outputs a binary classification - either *no damage (or only minor damage)*, or *major damage*. With this information, the relevant authorities could be notified and support can be provided immediately.

2 Related Work

Much of the previous work on this type of problem is done using convolutional neural networks (CNN). One method used Ordinal Regression in combination with CNN (Ci et al., 2019)[1]. Post-disaster building classification labels were used as ordinal variables (i.e. relative to other values on an arbitrary scale). Another approach comes from a group that trained three different CNNs on images at 3 different resolutions (100m, 10m, 5m) and used a combination of the information learned at each level for the final model. They used residual-connections and dilated convolutions for their models at each resolution (Duarte et al., 2018)[2]. Another approach used Single Shot Multibox Detector (Li et al., 2019)[3], which is a target detection method based on deep learning. This method identifies and outlines the buildings based on their level of damage.

Limitations for the previous work on this type of problem include using only high-definition images with well-defined crops of buildings, as well as using only post-disaster images from one type of natural disaster. As our project aims to use accessible satellite imagery enabling faster response times, the model cannot rely on having high-definition images as inputs. Furthermore, the project hopes that one algorithm can be applied to various types of natural disasters from different regions globally.

As we wanted to dedicate most of our time to the binary classification problem, we heavily relied on a Mask R-CNN project by Matterport[4] to help us implement Mask R-CNN (He et al., 2018) [5] to detect and outline buildings.

3 Dataset and Features

Our dataset is the xBD labeled dataset from xView2.org. The dataset is a collection of satellite images from before and after natural disasters. Almost all images contain buildings which are determined to be in one of 4 damage categories after the disaster (no damage, minor damage, major damage, or destroyed). The pre and post disaster images of each building are of virtually identical resolution and framed in a similar area in the image. Each building also comes "pre-outlined" within its image. The csv files containing the labels indicate polygons in which each building in an image is contained, helping to locate individual buildings in each image.

Data Statistics

	Train Data	Dev / Test Data
Number of images	5598	1866
Number of buildings (pre-disaster / post)	163819 / 162787	55224 / 54862
damage: none	117426 (72.13%)	41427 (75.5%)
damage: minor	14980 (9.2%)	4798 (8.75%)

[1] Tianyu Ci, Zhen Liu, Ying Wang (2019). Assessment of the Degree of Building Damage Caused by Disaster Using Convolutional Neural Networks in Combination with Ordinal Regression.
[2] Duarte, Diogo & Nex, Francesco & Kerle, Norman & Vosselman, George (2018). Satellite Image Classification Of Building Damages Using Airborne and Satellite Image Samples in a Deep Learning Approach.
[3] Yundong Li, Wei Hu, Han Dong, and Xueyan Zhang (2019). Building Damage Detection from Post-Event AerialImagery Using Single Shot Multibox Detector
[4] Abdulla, W. (2017). Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow. Github.
[5] He, Kaiming & Gkioxari, Georgia & Dollár, Piotr & Girshick, Ross (2018). "Mask R-CNN"

damage: major	14161 (8.7%)	3850 (7.02%)
damage: destroyed	13227 (8.13%)	3775 (6.88%)
damage: unclassified	2993 (1.84%)	1012 (1.85%)

Table1: Statistics on xBD labeled dataset.

As part of our pre-processing, we have removed all buildings classified as "unclassified." Furthermore, after human error analysis, we determined that buildings were too pixelated for us to accurately determine which class they fall in without a comparison to its pre-disaster image. Therefore, we decided to combine "no damage" and "minor damage" into one category and "major damage" with "destroyed" into another. We also noted that an overwhelming majority of the class distribution falls under "no damage," which could cause the model to become skewed. Hence, we further pre-processed the dataset to extract a subset containing an equal number of buildings from both classes.

To speed up training iterations, we trained our building classification model on 20,000 images and tested on 5,000. The input features are images with 64 by 64 pixels, which are normalized.

No damage Major damage No damage Major damage

Figure 1: Sample building images.

4 Methods

Our method follows a project pipeline (Figure 2) where an input image is fed into a Mask R-CNN model for object detection. This model outputs building coordinates on an image, which are then processed and cropped to feed to our binary classification model based on ResNet50[6].

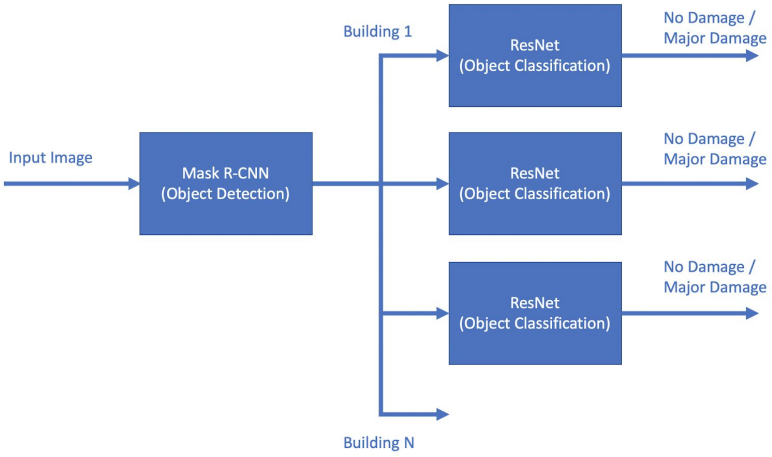

Figure 2: Project Pipeline

For building detection, we utilized Mask R-CNN, which is based on Feature Pyramid Networks (FPN) for object detection and a RESNET101 backbone. We preprocessed the satellite

[6] He, Kaiming & Zhang, Xiangyu & Ren, Shaoqing & Sun, Jian (2015). "Deep Residual Learning for Image Recognition"

imagery and used the coordinates in our training data to create masks for each of the buildings. We loaded the pre-trained weights for MS COCO. The model runs a binary classifier on a lot of anchors over the image and returns object/no-object scores. Positive anchors are defined as anchors that have an IoU of greater than 0.7. Subsequently, the model applies bounding box refinement, filters out low confidence detections, applies non-max suppression, and generates segmentation masks.

Satellite images and the bounding boxes for the buildings are then passed on to our second model to classify damages. Our damage classification model is based on the ResNet50 and is depicted in Figure 3. The model is a convolutional neural network with 50 hidden layers including a sigmoid layer as the output layer to perform the binary classification. Note that the sigmoid activation function, $\sigma = \frac{1}{1+e^{-x}}$ will output a probability that an example belongs to a given class, and so a decision can be made by comparing the probability with a given threshold. Each convolutional block and identity block consists of a series of convolutional layers, batch normalization, and ReLU layers, with a shortcut from the block's start to the block's end. Convolutional layers are utilized to train parameters arranged as filters more efficiently due to features such as parameter sharing and sparsity of connections. The shortcuts mitigate the vanishing gradient problem and allows for a deeper neural network, which enables more patterns to be learnt and hence improves the model's performance.

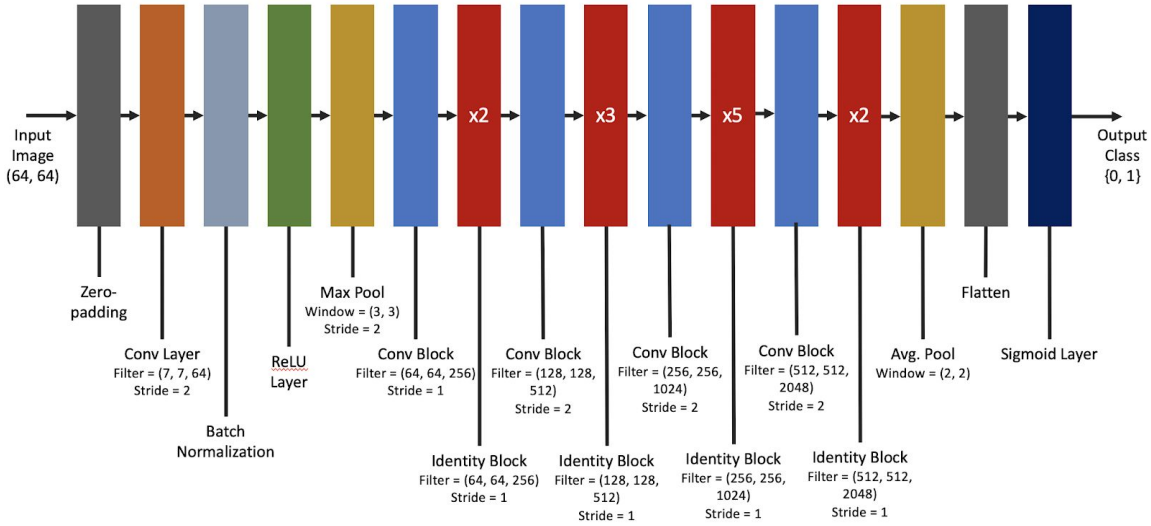

Figure 3: Damage Classification Model based on ResNet50.

5 Experiments / Results / Discussion

Initially, we performed a human error analysis for the damage classification task by having each member of the team manually classify 100 images as "damaged" or "not-damaged". We found that the group had an accuracy of 71% on average.

Subsequently, we built a baseline model for our damage classification problem. We used a neural network with two hidden layers, consisting of 50 units each and a ReLu activation function. We had initially wanted to classify the buildings into four labels and hence the final layer had four output nodes (no-damage, minor-damage, major-damage, destroyed). We split the training set into mini-batches of size 1,000 and chose to train a random 20,000 training images, with 5,000 testing images from the same distribution. Our model achieved a training set accuracy of 74.7% and a test accuracy of 72%. However, we used the unaltered version of the dataset, which was skewed toward no-damage (72%) and hence a model that always predicted no-damage would have performed similarly.

Following this, we switched to a ResNet50 model with 20,000 training images and 5,000 testing images. Furthermore, we improved our image pre-processing method in order to crop buildings of any size. We originally took 64x64 pixel crops of buildings by expanding out from their center, but this was not large enough for some buildings and caused 10% of our inputs to be incorrectly cropped. We adjusted to crop buildings of any size and would resize the larger images down to 64x64 pixels. On 20 epochs, this led to an accuracy of 84.6% on the training data and 53.9% on the testing data. On 50 epochs, the accuracy on the training data was 98.6% and the accuracy on the testing data was 77.9%, showing an improvement but indicating the model may have overfit.

To address the high variance, we added L2 regularization, trying both the activity regularizer and weights regularizer separately. The activity regularizer applies a penalty on the layer's output. With a hyper-parameter value of 0.01, training the model yielded an accuracy of 49.2% on the training set and an accuracy of 49.1% on the testing set. We then reduced the number of training images to 5,000, the number of testing images to 1,000, and the number of epochs to 20 in order to increase the iteration speed. We switched to a weights regularizer, applying a penalty on the weights trained, and tested different values for the regularization parameter (0.0001, 0.0003, 0.001, 0.01). The results indicate that a value of 0.0003 was optimum as anything less would not sufficiently reduce variance, but anything more significantly increased bias. With a value of 0.0003, the model achieved an accuracy of 72.1% on the training set, and an accuracy of 68.2% on the testing set. When using a larger data set (20,000 images for training and 5,000 images for testing), the accuracy on the training data was 87.5% and the accuracy on the testing data was 60%. Our best model's performance (68.2% accuracy on test set) is comparable to human performance (71% accuracy).

We also built a Mask R-CNN model to detect buildings from images. The resulting model, however, had very poor accuracy on the test set. We believe this is due to the fact that because of the large size of the training images, we were only able to train the model with 50 images and 18 epochs, which took longer than 10 hours on AWS with 8 GPUs. Even though the performance of our model was suboptimal, we believe that with adequate training, the model will achieve high accuracy rates.

6 Conclusion / Future Work

To conclude, although our results were inconclusive, the method remains promising. Despite the final model with an L2 regularization parameter of 0.0003 having accuracies short of the ResNet50 model without regularization, it may be a matter of finding the right parameter value. Moreover, upon closer inspection, we notice that with regularization, the model had a promising recall value of 79% on the test set, which may be the evaluation metric we want to focus on moving forward as it is arguably more important for buildings that are damaged to be identified correctly.

Error analysis was also performed in an attempt to find systematic errors and determine if any further image pre-processing can help. Roughly 20% of errors were made when the image contained a building heavily surrounded by trees. Another 16% of images may have contained buildings too pixelated or unclear that humans would also have trouble properly classifying the image. And another 15% contained multiple buildings within the cropped image. Based on this analysis, we could look into adding more images of buildings surrounded by trees to the training set. We could also work towards better image cropping techniques to mitigate the number of buildings per image.

As for building detection, we can look into other studies that have built models for the detection of buildings from satellite imagery. Perhaps, using the training weights for models specifically designed for building detection instead of those for general object detection will improve the accuracy of our model. Finally, other promising approaches include training on a bigger dataset and training the models longer.

7 Contributions

Each team member played an important role in this project. The following are some of the tasks handled by each of the team members

- Melvin: Preprocessing the images, writing the code for assessing human error, writing and improving the two models, debugging, and writing the reports.
- Cade: Preprocessing the images, human error assessment, error analysis, writing the reports, and creating the final video.
- T: Preprocessing the images, improving the models, error analysis, setting up and running all models on AWS, and writing the report.
- Abuzar: Preprocessing image annotations, writing the two models, error analysis, debugging, and contributing to the reports.

8 Code

The following is a link to our github repository: https://github.com/CX3XC/CS230
The building classification model can be found under Model_ResNet_final.py, with the ResNet50 infrastructure in Model_ResNet_helper.py. The building detection model can be found under Model pre_processing_mask_rcnn.ipynb

References

[1] Tianyu Ci, Zhen Liu, Ying Wang (2019). "Assessment of the Degree of Building Damage Caused by Disaster Using Convolutional Neural Networks in Combination with Ordinal Regression" https://www.mdpi.com/2072-4292/11/23/2858/htm

[2] Duarte, Diogo & Nex, Francesco & Kerle, Norman & Vosselman, George. (2018). Satellite Image Classification Of Building Damages Using Airborne and Satellite Image Samples in a Deep Learning Approach. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences. IV-2. 89-96. 10.5194/isprs-annals-IV-2-89-2018.

[3] Yundong Li, Wei Hu, Han Dong, and Xueyan Zhang (2019). "Building Damage Detection from Post-Event AerialImagery Using Single Shot Multibox Detector" https://www.mdpi.com/2076-3417/9/6/1128

[4] Abdulla, W. (2017). Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow. Github. https://github.com/matterport/Mask_RCNN

[5] He, Kaiming & Gkioxari, Georgia & Dollár, Piotr & Girshick, Ross (2018). “Mask R-CNN” https://arxiv.org/pdf/1703.06870.pdf

[6] He, Kaiming & Zhang, Xiangyu & Ren, Shaoqing & Sun, Jian. “Deep Residual Learning for Image Recognition” https://arxiv.org/pdf/1512.03385.pdf

[7] Ng, Andrew & Kian, Katanforoosh & Mourri, Younes. (Accessed 2020). Convolutional Neural Networks. Residual Networks Programming Assignment.