VECTOR MAP GENERATION FROM AERIAL IMAGERY USING DEEP LEARNING

M. Sahu* [1], A. Ohri [2]

[1] Indshine, New Delhi, manish@indshine.com
[2] Dept. of Civil Engineering, Indian Institute of Technology, (BHU) Varanasi, aohri.civ@iitbhu.ac.in

KEY WORDS: Building footprint, Segmentation, Aerial images, Vectorization, Deep Learning, GIS

ABSTRACT:

We propose a simple yet efficient technique to leverage semantic segmentation model to extract and separate individual buildings in densely compacted areas using medium resolution satellite/UAV orthoimages. We adopted standard UNET architecture, additionally added batch normalization layer after every convolution, to label every pixel in the image. The result obtained is fed into proposed post-processing pipeline for separating connected binary blobs of buildings and converting it into GIS layer for further analysis as well as for generating 3D buildings. The proposed algorithm extracts building footprints from aerial images, transform semantic to instance map and convert it into GIS layers to generate 3D buildings. We integrated this method in Indshine's cloud platform to speed up the process of digitization, generate automatic 3D models, and perform the geospatial analysis. Our network achieved ~70% Dice coefficient for the segmentation process.

1. INTRODUCTION

One of the major challenges in the GIS industry is the extraction of urban feature objects like buildings, roads, trees, etc. from Satellite and UAV images. Moreover, feature extraction is never an end product of any study rather it acts as an intermediate data from which analyses are done, so there is a need for a mechanism or a platform which can provide/generate data on the fly at industrial standard formats.

An increasing number of countries including developing countries are legalising UAVs and bringing supporting policies and regulations (Drone Laws By Country, 2018) for commercial use of drones. More tech companies like Planet Labs (Planet Labs, 2019), DJI (DJI, 2019) are enabling to capture such data very easily and safely. With the rise in the involvement of UAVs and Satellite data in construction (Drone Deploy, 2018), transportation (Indshine, 2019), and urban planning (Indshine, 2019), there is an ever-increasing demand for feature extraction methods. Manual features extraction using traditional GIS techniques is very time consuming and prone to errors.

Many pieces of research have focussed on segmentation using deep learning methods. xuran et.al. worked on Semantic segmentation using High-Resolution images and LIDAR data (Xuran Pan, 2018). Pascal et.al. used high-resolution images and OpenStreetMap data to segment images (Pascal Kaiser, 2017), number of competitions like ISPRS Semantic Labeling Contest (ISPRS, 2018) are also encouraging aerial image segmentation. Most of them focus only on semantic segmentation part instead of instance segmentation or vector generation.

This paper's objective is to combine powerful semantic segmentation algorithm with simple yet efficient image processing technique to generate instances, to fill gap between deep learning community and geospatial analyst community by proposing an automated pipeline to generate separated vector formats of connected binary blobs so that these can be used directly for advanced spatial analytics like calculating property taxes, estimating property loss during a disaster, verifying data for encroachment using land boundaries etc.

A GIS or a geospatial analyst needs to have instance objects in the vector formats to do critical analysis. For example, classified vectors can provide useful insights like the number of features, length, area, the perimeter of features, height (if available) etc.

Vectors can be stored easily in SQL databases to enable complex query analysis. Thus, we can say that producing only raster outputs, although very accurate, is not solving the real-world problems as it should be. That's why we are focusing on providing value to the end user by proposing simple semantic to instance segmentation and converting it into vector formats.

Many Deep Learning architectures like U-Net (Olaf Ronneberger, 2015), PSPNet (Hengshuang Zhao, 2017), DeepLabV3 (Liang-Chieh Chen, 2017) etc. are available for use. We used standard U-Net architecture in this paper with a slight addition of batch normalization layer after every convolution layer. However, our proposed method is independent of deep learning model architecture. To apply post-processing techniques, we went to the basics of image processing and used morphological operations to remove noises and used watershed segmentation to disjoint the connected blobs (Instances).

2. METHODOLOGY

In this section, we will present how post-processing pipeline is used with deep learning to enhance the segmentation results.

2.1 Semantic segmentation

For semantic segmentation of building footprint from RGB images, we used U-Net architecture. We added an extra batch normalization layer after every convolution layer to avoid any activations to take extreme values, to reduce sensitivity towards initial weights initialization and reduce overfitting of the model.

* Corresponding author

(Ronneberger et al, 2015) describes all the details of architecture and used it for medical image segmentation. In the original U-Net, Olaf used 512x512 image size in the network but here we have used 200x200 image size. Figure 2.1 describes the U-Net architecture.

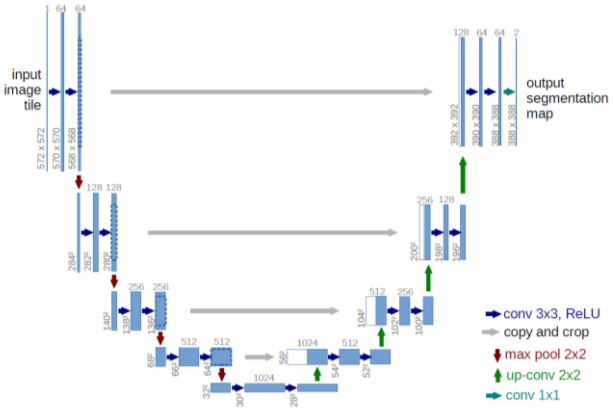

Figure 2.1: U-Net deep neural network architecture.
Source: Olaf Ronneberger, 2015

2.2 Training Procedures

The input images and their segmentation (binary) images were used to train the network with the Adam optimizer (Kingma, 2014). Since all the images were georeferenced, we used open source Gdal library (Contributors of GDAL/OGR, 2018) to pre-process our data before feeding it to the neural network. Input images are of very large size, thus, they were divided into 200x200 grids. Both inputs and labels were given the same transformation in order to avoid any misalignment. We used 300 epochs and batch size of 16 while training and testing. We used Dice coefficient (Dice, 1945) as our loss function. Whole training was done on Ryzen 1700x, 32GB RAM CPU and GTX 1060 6GB GPU.

2.3 Training datasets

For training dataset, we used aerial images in color band from 3 sources.

1. Inria aerial dataset, Structured Buildings (Emmanuel Maggiori, 2017)
2. Massachusetts building dataset, Structured Buildings (Mnih, 2013)
3. Indshine's UAV orthomosaic dataset of Maharashtra, India region of about ~100 square km

Building in training dataset of Inria and Massachusetts are very structured and not densely built whereas in India buildings are unstructured, and the density of buildings is very high. So, the model trained only on datasets of the likes of Inria and Massachusetts will never work in conditions like India where density is high and buildings are convoluted.

Structured buildings are those which are uniformly spaced. Unstructured buildings are referred to those buildings which are placed randomly and are built at very closed distances without following any pattern.

Source of dataset	Resolution/ GSD	Type of dataset
Inria Aerial dataset	30 cm	Satellite (Public)
Massachusetts building dataset	100 cm	Satellite (Public)
UAV's orthomosaics	50 cm	UAV (Private)

Table 2.1: Sources of dataset and GSD (Ground Sampling Distance)

Type of dataset from UAV orthomosaics	Number of Images (200x200) / Area
Training	8000/ 80 sq. Km
Testing	2000/ 20 sq. Km

Table 2.2: Training and Testing dataset in case of Private orthomosaics

2.4 Post-Processing

Output obtained from U-Net is in binary format, where each pixel represents the building or background class.

Since raster data of buildings is not of much use for geospatial analyst community, we followed a post-processing pipeline. Post-processing takes raster as input and produces output in vector format after removing the noises and separating connected building vectors.

2.5 Noise removal

For noise removal, we used basic morphological operation on binary data. We first applied erosion followed by dilation followed by thresholding with respect to area. This removed small noises from our output as shown Figure 2.2

Figure 2.2: Represents noise removal output

2.6 Distance Transform

To separate connected buildings, we assumed that connections/binary bridge in the joined binary blobs is less than the area of buildings itself. We used this fact and applied distance transformation (Jain, 1989) to binary image. Thus connection/binary bridge were assigned less weight as compared to buildings itself. See Figure 2.3 for illustration.

We thought of using thresholding to remove this connection, but this gave us very poor results. It was because distance transform assigns more weight to large blobs and less weight to smaller blobs and edge pixels. So, for large connected buildings, even the weight of connection/binary bridge was greater than that of small buildings weights itself. Thus, applying thresholding not only removed connections but also small buildings. Refer to Figure 2.4 for distance transform output.

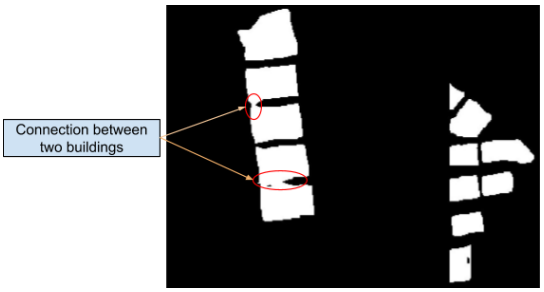

Figure 2.3: Connection/binary bridge between buildings

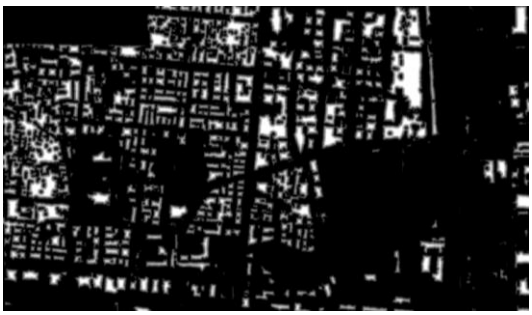

Figure 2.4: Distance transform output

2.7 Local Maxima

To overcome the problem described above, we used Local Maxima approach. Since connections are smaller than the buildings itself, finding local maxima ensured that it lies inside building area and not in the connection. This local maximum point acted as input source/sink to the watershed algorithm.

2.8 Watershed Segmentation

Watershed segmentation is quite popular for image segmentation. Local Maxima obtained acted here as sink point and negative of distance transform as cost map. This helped to separate connected binary blobs effectively. Each blob is given a unique index for further processing, shown in Figure 2.5.

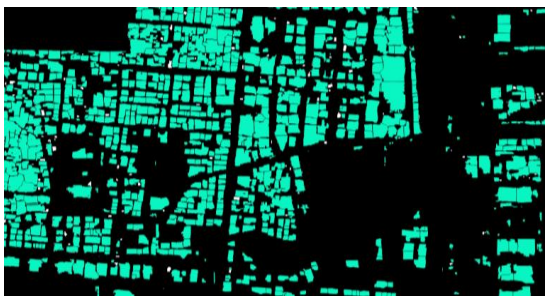

Figure 2.5: Watershed Segmentation output

2.9 Vectorization, Smoothing and Minimum Bounding Box Estimation

Raster obtained from the watershed segmentation is vectorized using gdal/ogr library (Contributors of GDAL/OGR, 2018). Spatial references and coordinate systems are preserved at every step and are transferred to vector file for correct overlaying.
While converting from raster to vector, the output has a lot of vertices and noises. Vector is then simplified using Douglas-Peucker algorithm (David Douglas, 1973). This helps to preserve the overall geometry of the shape while simplifying number of vertices. Basic attributes like area, perimeter and elevation of the buildings were automatically added. Finally, minimum bounding box was used and saved. Refer to Figure 2.6 to see the separated vector of closely connected buildings, polygon simplification.

Figure 2.6: Red- Noisy vector, Blue- Simplified vector
Green- Minimum Bounding Box

3. CHALLENGES

The primary challenges that we faced while working on this problem were

1. To make a single model for both structured and unstructured building features
2. To separate densely compacted unstructured buildings
3. To make this method industry-ready and usable by providing data in required formats via an online platform

3.1 Our Approach

3.1.1 To make a single model for both structured and unstructured building features: When we have very distinct datasets (here structured and unstructured), sequential training (training one after another) doesn't help. This kind of training procedure will make the model forget features learned in step1. We trained a model of structured dataset and took it as a base model. Then we used unstructured building dataset and a part of the structured building dataset (let us call it Mixed dataset) as training images and started training. This approach helped us keep both types of building features. Refer to accuracy section in this document to see effect of mixed dataset learning.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-IV-2-W5-157-2019 | © Authors 2019. CC BY 4.0 License.

3.1.2 To separate densely compacted unstructured buildings: We followed post-processing pipeline, as explained above, to separate densely compacted buildings.

3.1.3 To make method industry ready and usable by providing data in required formats and via online platform: We integrated this algorithm to Indshine's cloud platform (Indshine, 2018). It has got multiple organization's user base where people often collaborate and do digitization, geospatial analysis, extract design information. All the vector data were overlaid over high-resolution orthomosaic map.

4. RESULTS

We have tested our model both on the structured and unstructured datasets separately. Accuracy in structured dataset was slightly higher than unstructured dataset. Before training on the Mixed dataset, this model gave an accuracy of 43.75% whereas after training on the Mixed dataset, the we achieved an accuracy of 69.62%, significantly much higher.

Type of Training	Dice Coefficient (%)
Model untrained on mixed dataset	43.75
Model trained on mixed dataset	69.62

Table 1: Accuracy

5. CONCLUSIONS

5.1 Semantic to Instance Segmentation: Semantic to Instance segmentation can be made using simple post-processing pipeline. These are highly effective in the segmentation of structured building datasets or low to medium density buildings. For areas of very high-density buildings as in Figure 8.3 and 8.4, our model could identify large part of the building features as well as separate them with slightly lesser accuracy. However, it was very difficult to separate small buildings with large connecting areas.

5.2 Common model for both structured and un-structured datasets: We could easily capture relatively distinct feature from a single model by using mixed dataset. We increased our accuracy by nearly 30% using mixed dataset.

5.3 Ready to use GIS layers: Our model produces an output of simplified vector data. This is widely used by GIS analysts to perform spatial analysis. Few use cases are given below under the Use Cases section. Integrating such algorithms with a web platform can prove to be highly helpful for GIS community. In fact, generating vector data on the web platform will also allow us to automatically generate 3D models.

6. USE CASES

6.1 Faster/Automatic Digitization: By digitization, we mean the process of extracting the important features from a digital image, with all the features having geographic coordinates associated with it. This is an important technique for data input and storage in a GIS environment. However, the current process of manual digitization is extremely time-consuming and expensive where people, for example, draw polygons for each building or cluster of buildings, trees and roads. The number of buildings alone can be in the order of 100,000, and it becomes increasingly difficult for people to mark all those buildings one by one. Therefore, if the proposed algorithm is able to increase the productivity of the people involved in digitization by even 30%, we can save a lot of time and resources as well as enable the industry to use the data further. In the first phase, this algorithm along with collaboration with Indshine's cloud platform can be used in such a way that if the user clicks on any building, it will prompt the estimated footprint (in the form a polygon) of that building with all the vertices of the polygon representing the corners of the building. This way, the user can behave in a reactive instead of a proactive worker.

6.2 Property Tax Estimation: Taxes serve as a major source of revenue for any Government, and the property tax is one of them. The municipality or the local government finds it difficult to estimate the amount of property taxes to be collected from a city without a proper system. Property tax depends on three major parameters namely property type (commercial or residential), area, and the number of floors in the property. We propose to estimate the area of the property using our algorithm, as well as the number of floors based on the elevation models of the building since elevation and number of floors are correlated. Thus, the government body, after ground survey for classifying into commercial and residential property, can use our models for the tax estimation, monitoring, and analysis.

6.3 Disaster Impact Analysis: Use of satellites and UAVs are increasingly becoming popular for disaster response analysis. It is very crucial to get the disaster impact map in order to prioritize the rescue operations. Usually, it is very difficult and time-consuming for a team to manually identify the condition of buildings and roads post-disaster. Hence, it becomes very crucial to create damage map accurately and immediately. Deep neural networks can help in this process by automatic detection of features like buildings and roads. Our algorithm can be used directly by the disaster response team to quickly extract all the buildings in that area pre-disaster and post-disaster. Thereafter, they can identify the specific regions of destruction by comparing between the pre-disaster and post-disaster images. (Jigar Doshi, 2018) from Facebook research and CrowdAI recently showed us a very useful technique to create Disaster Impact Index (DII) to estimate the disaster impact.

7. LIMITATIONS

Post-Processing not using RGB: One of the crucial limitations of this post processing pipeline is that we have not considered RGB data while separating connected buildings. We worked on the fact that connections are generally smaller than buildings itself. This assumption is sometimes not true when buildings are small and very dense. This is applicable for slums and this case illustrated in Figure 7.1 and Figure 7.2. So here we are losing out an extra information

Figure 7.1 Ineffective separation of buildings due to large contact area

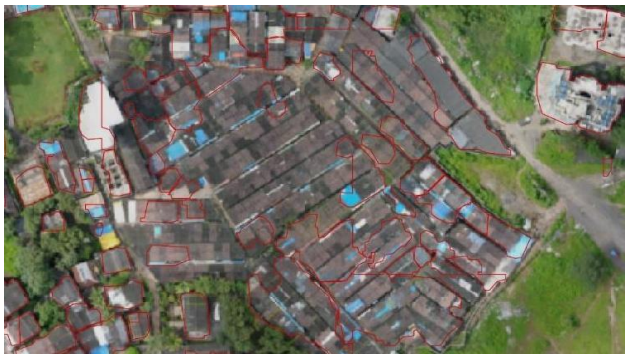

Figure 7.2 Model fail to detect buildings here due to poor building textures

8. VISUALIZATION OF RESULTS

Figure 8.1 Dataset from Boston (Mnih, 2013)

Structured building dataset taken from Satellite of Boston Area. Resolution of the image is 30cm

Figure 8.2 Dataset from Maharashtra (Indshine, 2018)

Unstructured building dataset taken from UAV of village in Maharashtra. Resolution of image is 50cm

Figure 8.3 Dataset from Tanzania (Source: OpenAerialMaps)

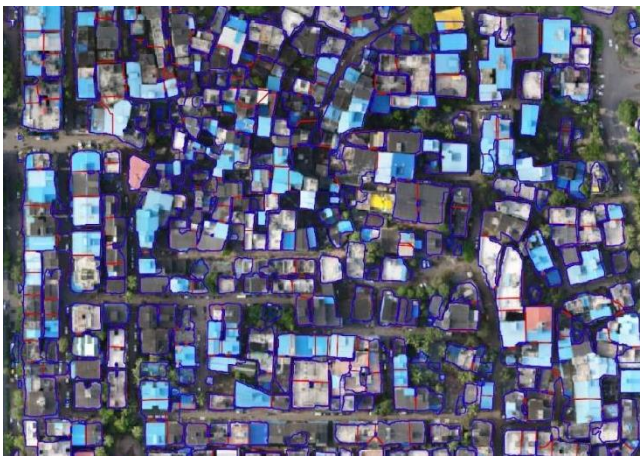

Figure 8.4 Dataset from Mumbai (Indshine, 2018)

Unstructured building dataset taken from UAV near Mumbai slums region. Blue boundary is raw detection without post-processing. Red boundary represents where polygon is divided into parts.

9. ACKNOWLEDGEMENTS

I would like to thank Prince Diwakar from Indshine for helping me prepare manuscript, and whole Indshine team for providing me resources to generate label dataset and train my model.

10. REFERENCES

Contributors of GDAL/OGR. (2018). Geospatial Data Abstraction Software Library.

David Douglas, T. P. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. The Canadian Cartographer 10(2), doi:10.3138/FM57-6770-U75U-7727, 112–122.

DeepGlobe. (2018). Challenges. From deepglobe.org: http://deepglobe.org/challenge.html

Dice, L. R. (1945). Measures of the Amount of Ecologic Association Between Species. Ecology. doi:10.2307/1932409. JSTOR 1932409., 297-302.

DJI. (2019). DJI:Future of Drones. From DJI: https://www.dji.com/products/drones

Drone Deploy. (2018, July 07). The Rise of Drones in Construction. From Drone Deploy: https://blog.dronedeploy.com/the-rise-of-drones-in-construction-5357b69942fa

Drone Laws By Country. (2018). From uavsystemsinternational: https://www.uavsystemsinternational.com/drone-laws-by-country

Emmanuel Maggiori, Y. T. (2017). Can Semantic Labeling Methods Generalize to Any City? The Inria Aerial Image Labeling Benchmark. IEEE International Geoscience and Remote Sensing Symposium (IGARSS).

Hengshuang Zhao, J. S. (2017). Pyramid Scene Parsing Network.

Indshine. (2018). Indshine's Cloud platform. From Indshine: Drone analytics platform: www.indshine.com

Indshine. (2019, Januray 08). Drones help save lives by reducing railway accidents. From www.indshine.com: https://medium.com/indshine/drones-help-save-lives-by-reducing-railway-accidents-861013c51e04

Indshine. (2019, January 8). Railways: Indshine mapped 263 Km for the construction of the 3rd railway line. From www.Indshine.com: https://medium.com/indshine/railways-how-indshine-helped-indian-railways-by-mapping-263-km-to-construct-3rd-railway-line-84b8e9986319

ISPRS. (2018). 2D Semantic Labeling Contest. From ISPRS.org: http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html

Kingma, D. P. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Liang-Chieh Chen, G. P. (2017). Rethinking Atrous Convolution for Semantic Image Segmentation.

Mnih, V. (2013). Machine Learning for Aerial Image Labeling. University of Toronto.

Olaf Ronneberger, P. F. (2015). U-Net: Convolutional Networks for Biomedical.

Pascal Kaiser, J. D. (2017). Learning Aerial Image Segmentation. IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, Volume: 55 Issue: 11 , 6054 - 6068.

Planet Labs. (2019). Planet labs. From Planet labs: https://www.planet.com

Xuran Pan, L. G. (2018). Semantic Labeling of High Resolution Aerial Imagery. MDPI Remote Sensing, 10, 743.

Jigar Doshi, Saikat Basu, Guan Pang (2018). From Satellite Imagery to Disaster Insights, AI for Social Good Workshop at NeurIPS 2018