Map Generation from Large Scale Incomplete and Inaccurate Data Labels

Rui Zhang
rui.zhang@ibm.com
IBM T.J. Watson Research Center
Yorktown Heights, New York

Conrad Albrecht
cmalbrec@us.ibm.com
IBM T.J. Watson Research Center
Yorktown Heights, New York

Wei Zhang
weiz@us.ibm.com
IBM T.J. Watson Research Center
Yorktown Heights, New York

Xiaodong Cui
cuix@us.ibm.com
IBM T.J. Watson Research Center
Yorktown Heights, New York

Ulrich Finkler
ufinkler@us.ibm.com
IBM T.J. Watson Research Center
Yorktown Heights, New York

David Kung
kung@us.ibm.com
IBM T.J. Watson Research Center
Yorktown Heights, New York

Siyuan Lu
lus@us.ibm.com
IBM T.J. Watson Research Center
Yorktown Heights, New York

Figure 1: Houses (red semi-transparent boxes) and roads (white semi-transparent lines) generated from aerial imagery (background).

ABSTRACT

Accurately and globally mapping human infrastructure is an important and challenging task with applications in routing, regulation compliance monitoring, and natural disaster response management etc.. In this paper we present progress in developing an algorithmic pipeline and distributed compute system that automates the process of map creation using high resolution aerial images. Unlike previous studies, most of which use datasets that are available only in a few cities across the world, we utilize publicly available imagery and map data, both of which cover the contiguous United States (CONUS). We approach the technical challenge of inaccurate and incomplete training data adopting state-of-the-art convolutional neural network architectures such as the U-Net and the CycleGAN to incrementally generate maps with increasingly more accurate and more complete labels of man-made infrastructure such as roads and houses. Since scaling the mapping task to CONUS calls for parallelization, we then adopted an asynchronous distributed stochastic parallel gradient descent training scheme to distribute the computational workload onto a cluster of GPUs with nearly linear speed-up.

KEYWORDS
remote sensing, geo-spatial analysis, image segmentation, aerial image processing, map generation, deep neural networks, U-Net, CycleGAN
ACM Reference Format:
Rui Zhang, Conrad Albrecht, Wei Zhang, Xiaodong Cui, Ulrich Finkler, David Kung, and Siyuan Lu. 2020. Map Generation from Large Scale Incomplete and Inaccurate Data Labels. In KDD ’20: ACM SIGKDD Conference on Knowledge Discovery and Data Mining, August 22–27, 2020, San Diego, CA. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnn.nnnnnn

1 INTRODUCTION

Generating maps of roads and houses from high resolution imagery is critical to keep track of our ever changing planet. Furthermore, such capability become vital in disaster response scenarios where pre-existing maps are often rendered useless after destructive forces struck man-made infrastructure. New maps of roads accessible as well as indication of destroyed buildings will greatly help the disaster response team in rescue planning.

Due to the significant advancement in computer vision by deep learning during the last couple of years in parallel to the explosive amount of high-resolution imagery becoming available, there has been a growing interest in pushing research in automated map generation from high resolution remote sensing imagery [2, 11, 19]. Several challenges in the field of detecting houses or roads using high-resolution remote sensing data have been established–two of them being SpaceNet [2] and DeepGlobe [5]. SpaceNet hosted a series of challenges including three rounds of building detection competitions, and two rounds of road network detection contests. The SpaceNet building detection challenge provides 5 cities (plus one additional city–namely Atlanta, GA, USA–in the last round) across different continents, including Las Vegas, NV, USA; Paris, France; Rio de Janeiro, Brasil; Shanghai, China; and Khartoum, Sudan. The corresponding building and house labels are provided along with the high resolution imagery. The DeepGlobe challenge was using the same set of data, but employed a slightly different evaluation measure.

The SpaceNet and DeepGlobe training datasets being publicly available attracted researchers globally to address the challenge of automated map generation. However, as is commonly known infrastructure vary significantly from one location to another. While high-rise apartment buildings have become the typical in China whereas most US population lives in single family houses. A model trained on US data will perform poorly on China. Even within the US from state to state the geo-spatial variation is evident. From this perspective a task of training a model to generate a global map requires more geographically diverse training data. Indeed, the past decade has shown a dramatic increase in the amount of open geo-spatial datasets made available by government agencies such as the United States Department of Agriculture (USDA), NASA, the European Space Agency. Specifically in this paper, we employ two publicly available datasets to train the model of map generation from imagery. One from the corpus of OpenStreetMap (OSM) data and another from the National Agriculture Imagery Product (NAIP) distributed by USDA.

Most state-of-the-art automated programs rely on a significant amount of high-resolution image data with a correspondingly well labeled map [11, 17]. There has been limited studies utilizing the freely available OSM due to its fluctuating accuracy depending on the OSM community’s activity in a given geo-location. In this work we attempt to utilize the inaccurate and incomplete labels of OSM to train neural network models to detect map features beyond OSM.

In addition to the challenge in data quality, the data volume needed to train a model that can map the whole United States is daunting too. The total area of the CONUS is about 7.7 million square kilometers. For NAIP imagery of 1 m resolution, the amount of data sums up to about 44 TB. In view of the very large scale training dataset, we adopted an asynchronous distributed parallel stochastic gradient descent algorithm to speed up the training process nearly linearly on a cluster of 16 GPUs.

We propose a training framework to incrementally generate maps from inaccurate and incomplete OSM data with high resolution aerial images, using data from four cities in Texas drawn from OSM and NAIP. Moreover, Las Vegas data from SpaceNet have been employed as well. The salient contribution is the following:

- We propose a training frame work to incrementally generate map from inaccurate and incomplete OSM maps.
- To our best knowledge, it is the first time OSM data is employed as a source of rasterized map imagery in order to train an pixel-to-pixel mapping from high resolution imagery.
- We analyzed the completeness of the OSM data and discuss an evaluation metric given the inaccuracy and incompleteness of the OSM labels.
- We numerically investigate the transfer of a model from one geo-location to another to quantify how well the models under consideration generalize.
- We tested an asynchronous parallel distributed stochastic gradient descent algorithm to speed up the training process nearly linearly on a cluster of 16 GPUs.
- Through a publicly accessible tool we showcase how we make available the our generated maps.

Figure 2: Sample OSM data near Dallas, TX and San Antonio, TX: The plot on the upper left shows an urban area well labeled with most roads (white/yellow lines) and houses (red boxes) accurately marked. The sample on the upper right is drawn from a rural location with less accurate labels. The lower left plot represents another rural region with partially labeled houses. Finally, the lower right figure exemplifies a highly populated area without any house record in the OSM dataset, yet.
2 RELATED WORK

Employing remote sensing imagery to generate maps has become more popular due to the rapid progress in deep neural network in the past decade—particularly in the arena of computer vision.

- encoder–decoder–type convolutional neural networks for population estimation from house detection [25], or road detection with distortion tolerance [31], or ensembles of map-information assisted U-Net models [21]
- ResNet-like down-/upsampling for semantic segmentation of houses: [19]
- image inpainting for aerial imagery for semi-supervised segmentation: [24]
- human infrastructure feature classification in overhead imagery with data prefiltering: [3]

Distributed deep learning is the de-facto approach to accelerate its training. Until recently, it was believed that the asynchronous parameter-server-based distributed deep learning method is able to outperform synchronous distributed deep learning. However, researchers demonstrated synchronous training is superior to asynchronous parameter-server based training, both, from a theoretical and an empirical perspective [8, 9, 30]. Nevertheless, the straggler problem remains a major issue in synchronous distributed training, in particular in a large scale setting. Decentralized deep learning [13] is recently proposed to reduce latency issues in synchronous distributed training and researchers demonstrated that decentralized deep learning is guaranteed to have the same convergence rate as synchronous training. Asynchronous decentralized deep learning is further proposed to improve runtime performance while guaranteeing the same convergence rate as the synchronous approach [14]. Both scenarios have been verified in theory and practice over 100 GPUs on standard computer vision tasks such as ImageNet.

3 DATA CORPUS

Since geo-spatial data comes in different geo-projections, spatial and temporal resolution, it is critical to correctly reference it for consistent data preparation. In this study, we used three sources of data: NAIP, OSM, and SpaceNet. To obtain consistency in the generated training data, a tool developed by IBM was utilized: PAIRS [12, 15], shorthand for Physical Analytics Integrated data Repository and Services. The following sections describe in detail how PAIRS processes each dataset to generate a uniform, easily accessible corpus of training data.

3.1 Big Geo-Spatial Data Platform

Both the NAIP imagery and OSM rasterized data are first loaded into IBM PAIRS. PAIRS is a big spatio-temporal data platform that curates, indexes and manages a plurality of raster and vector datasets for easy consumption by cross-datalayer queries and geospatial analytics. Among others, datasets encompass remote sensing data including satellite/aerial images, point cloud data like LiDAR (Light Detection And Ranging) [27], weather forecast data, sensor measurement data (cf. Internet of Things [10]), geo-survey data, etc.

PAIRS [12, 16] is based on the scalable key-value store HBase[26]. For users it masks various complex aspects of the geo-spatial domain where e.g. hundreds of formats and geo-projections exist—provided by dozens of data sources. PAIRS employs a uniform geo-projection across all datasets with nested indexing (cf. QuadTree [7]). Raster data such as satellite images are cut into cells of size 32 by 32 pixels. Each cell is stored in HBase indexed by a 16 bytes key encoding its spatial and temporal information. Among others, the key design ensures efficient data storage and fast data retrieval.

3.2 Training Data Characteristics

USDA offers NAIP imagery products which are available either as digital ortho quarter quad tiles or as compressed county mosaics [20]. Each individual image tile within the mosaic covers a 3.75 by 3.75 minute quarter quadrangle plus a 300 meter buffer on all four sides. The imagery comes with 4 spectral bands covering a red, green, blue, and near-infrared channel. The spatial resolution effectively varies from half a meter up to about two meters. The survey is done over most of the territory of the CONUS such that each location is revisited about every other year.

The open-source project OpenStreetMap (OSM) [1] is a collaborative project to create a freely available map of the world. The data of OSM is under tremendous growth with over one million registered contributors editing and updating the map. On the one hand, OSM data stem from GPS traces collected by voluntary field surveys such as e.g. hiking and biking. On the other hand, a web-browser–based graphical user interface with high resolution satellite imagery provides contributors across the world an online tool to generate vector datasets annotating and updating information on roads, buildings, land cover, points of interest, etc.

Both approaches have limitations. Non-military GPS is only accurate within meters—roads labeled by GPS can be off by up to 20 meters. To the contrary, manual annotation is time consuming and typically focused on densely populated areas like cities, towns, etc. Fig. 2 shows examples of OSM labels with satellite imagery as background for geo-spatial reference. By visual inspection, it is evident that in most of the densely populated areas, roads are relatively well labeled. Concerning houses the situation is similar. However, compared to surrounding roads, more geo-spatial reference points need to be inserted into the OSM database per unit area in order to label all buildings. Thus, its coverage tends to lag

Figure 3: Sample training data generated from PAIRS retrieval. The curation and geo-indexing of the PAIRS system aligns the aerial imagery and the rasterized OSM map to exactly match pixel by pixel.

PAIRS [12, 16] is based on the scalable key-value store HBase[26]. For users it masks various complex aspects of the geo-spatial domain where e.g. hundreds of formats and geo-projections exist—provided by dozens of data sources. PAIRS employs a uniform geo-projection across all datasets with nested indexing (cf. QuadTree [7]). Raster data such as satellite images are cut into cells of size 32 by 32 pixels. Each cell is stored in HBase indexed by a 16 bytes key encoding its spatial and temporal information. Among others, the key design ensures efficient data storage and fast data retrieval.

3.2 Training Data Characteristics

USDA offers NAIP imagery products which are available either as digital ortho quarter quad tiles or as compressed county mosaics [20]. Each individual image tile within the mosaic covers a 3.75 by 3.75 minute quarter quadrangle plus a 300 meter buffer on all four sides. The imagery comes with 4 spectral bands covering a red, green, blue, and near-infrared channel. The spatial resolution effectively varies from half a meter up to about two meters. The survey is done over most of the territory of the CONUS such that each location is revisited about every other year.

The open-source project OpenStreetMap (OSM) [1] is a collaborative project to create a freely available map of the world. The data of OSM is under tremendous growth with over one million registered contributors editing and updating the map. On the one hand, OSM data stem from GPS traces collected by voluntary field surveys such as e.g. hiking and biking. On the other hand, a web-browser–based graphical user interface with high resolution satellite imagery provides contributors across the world an online tool to generate vector datasets annotating and updating information on roads, buildings, land cover, points of interest, etc.

Both approaches have limitations. Non-military GPS is only accurate within meters—roads labeled by GPS can be off by up to 20 meters. To the contrary, manual annotation is time consuming and typically focused on densely populated areas like cities, towns, etc. Fig. 2 shows examples of OSM labels with satellite imagery as background for geo-spatial reference. By visual inspection, it is evident that in most of the densely populated areas, roads are relatively well labeled. Concerning houses the situation is similar. However, compared to surrounding roads, more geo-spatial reference points need to be inserted into the OSM database per unit area in order to label all buildings. Thus, its coverage tends to lag

1 as of February 2020, cf. https://wiki.openstreetmap.org/wiki/Stats
behind road labeling. Shifting attention to rural and remote areas, the situation becomes even worse, because there is less volunteers available on site—or there is simply less attention and demand to build an accurate map in such geographic regions.

Since our deeply learnt models are conceptually based on a series of convolutional neural networks to perform auto-encoding–type operations for the translation of satellite imagery into maps, the problem under consideration could be rendered in terms of image segmentation. Therefore, we pick a rasterized representation of the OSM vector data, essentially projecting the information assembled by the OSM community into an RGB-channel image employing the Mapnik framework [18]. This way each color picked for e.g. roads (white), main roads (yellow), highways (blue), etc. and houses (brown) becomes a unique pixel segmentation label.

The rasterized OSM data is then geo-referenced and indexed in PAIRS such that for each NAIP pixel there exists a corresponding unique label. The label generation by the OSM rasterization is performed such that the OSM feature with the highest z-order attribute2 is picked. E.g. if a highway crosses a local road by bridge, the highway color label is selected for the RGB image rasterization. This procedure resembles the top-down view of the satellite capturing the NAIP imagery. A notable technical aspect of PAIRS is its efficient storage: Besides data compression, no-data pixels (cf. the GeoTiff standard [4]) are not explicitly stored. In terms of uploading the rasterized OSM data, this approach significantly reduced the amount of disk space needed. Per definition, we declare the OSM sandy background color as no-data.

After both NAIP and rasterized OSM data is successfully loaded into PAIRS, we use Apache Spark [28] to export the data into tiles of 512 × 512 pixels. The uniform geo-indexing of the PAIRS system guarantees aerial imagery and rasterized maps to exactly match at same resolution and same tile size. An illustration of training samples shows Fig. 3. For this study, we focused on the state of Texas, where we exported a total of about 2.8 million images with total volume of 2.6 TB. However, we limit our investigation to the big cities in Texas with rich human infrastructure, namely: Dallas, Austin, Houston, and San Antonio.

3.3 SpaceNet: Data Evaluation Reference

Since OSM data bear inaccuracies and it is incomplete in labeling, we picked a reference dataset to test our approach against a baseline that has been well established within the last couple of years within the remote sensing community. SpaceNet offers e.g. accurately labeled building polygons for the city of Las Vegas, NV. Given a set of GeoJSON vector data files, PAIRS rasterized these into the same style as employed by the OSM map discussed above. Then, the corresponding result is curated and ingested as a separate raster layer, ready to be retrieved from PAIRS the same way as illustrated in Fig. 3. This setup allows us to apply the same training and evaluation procedure to be discussed in the sections below.

4 MODELS

In this study we have used two existing deep neural network architectures with further improvement on the baseline models in view of the characteristics of the data available (cf. Sect. 1). Details of the models employed are described in the following.

4.1 U-Net

U-Net [22] is a convolutional neural network encoder-decoder architecture. It is the state-of-the-art approach for image segmentation tasks. In particular, U-Net is the winning algorithm in the 2nd round of the SpaceNet [6] challenge on house detection using high resolution satellite imagery. The U-Net architecture consists of a contracting path to capture context, and employs a symmetric expanding path that enables precise localization. The contracting path is composed of a series of convolution and max-pooling layers to coarse-grain the spatial dimensions. The expanding path uses up-sampling layers or transposed convolution layers to expand the spatial dimensions in order to generate a segmentation map with same spatial resolution as the input image. Since the expanding path is symmetric to the contracting path with skip connections wiring, the architecture is termed U-Net. Training the U-Net in a supervised manner for our remote sensing scenario requires a pixel-by-pixel–matching rasterized map for each satellite image.

4.2 CycleGAN

The CycleGAN model was introduced in 2017 [32] for the task of image-to-image translation which is a class of computer vision problems with the goal to learn the mapping between two distributions of unpaired data sets \( X = \{ x \} \) and \( Y = \{ y \} \). Given images \( x \) from a source distribution \( p_{\text{data}}(x) \) and maps \( y \) from a target distribution \( p_{\text{data}}(y) \), the task is to learn a mapping \( G : X \rightarrow Y \) such that the distribution of \( G(x) \) is as close as possible to the distribution of \( p_{\text{data}}(y) \). In addition, a mapping \( F : Y \rightarrow X \) is established to further regulate the network’s learning by a so called cycle consistency loss enforcing \( F(G(x)) \approx x \). Starting off with \( y \) and repeating this line of reasoning, a second cycle consistency loss pushes the CycleGAN’s numerical optimization towards \( G(F(y)) = y \).

The paper introducing CycleGAN [32] provided a showcase that translated satellite imagery to maps by way of example. Some rough, qualitative measurement on CycleGAN’s ability to convert overhead imagery to maps was provided. In our paper, it is the first time CycleGAN is evaluated quantitatively in terms of house and road detection.

For the discussion to follow, \( x \in X \) represents imagery input, \( y \in Y \) map images. The corresponding CycleGAN generators are \( \hat{x} = G_X(y) \) and \( \hat{y} = G_Y(x) \), respectively.

4.3 Feature-Weighted CycleGAN

In generating maps from aerial images, the focus is to precisely extract well defined features such as e.g. houses and roads from the geographic scene. Thus, we added one more loss to the CycleGAN’s

Figure 4: Two pairs of samples of rasterized OSM maps (RGB color, left) with their corresponding feature mask (bi-color, right) next to each other.
Figure 5: Given satellite imagery data $x$ and corresponding map data $y$: illustration of our CycleGAN architecture showing the data flow from image input $x$, to generated map $\hat{y}$, to recreated image $\hat{x} = G_X(\hat{y})$. This cycle is used to compute a consistency loss $\|x - \hat{x}\|$ which is weighted by the feature map $M(y)$ yielding the FW-loss contribution during training.

Figure 6: Sample scenario from which to compute the $F_1$ score: On the top row (a), (b) and (c) are the OSM map $y$, the generated map $\hat{y}$, and the NAIP image $x$, respectively. On the bottom row, (d), (e) and (f) are houses as labeled by the OSM map and houses detected. It results in the following cases colored: correctly detected houses (TP, red), missed houses (FN, pink), and false positive houses (FP, cyan).

5 EVALUATION METRICS

We adopted a feature-level detection score, similar to the SpaceNet evaluation approach: Each house or road detected by the generated map $\hat{y}$ is evaluated against the OSM map $y$ using a binary classification score $F_1$ which consists of both, precision and recall. In the following section we detail on how each score is computed.

In a first step, detected feature pixels in the maps (both, OSM map $y$ and generated map $\hat{y}$) are extracted using the same method as described in Sect. 4.3. Then a set of polygons $\{P^y_i\} \cup \{P^\hat{y}_j\}$ is
generated from the extracted features. A feature like a house in the real map $y$ represented by a polygon $P^y_i$ is correctly detected if there exists a corresponding polygon $P^\hat{y}_j$ in the generated map $\hat{y}$, such that the intersection over union (IoU)

$$\text{IoU}_{ij} = \frac{|P^\hat{y}_j \cap P^y_i|}{|P^\hat{y}_j \cup P^y_i|} \quad (3)$$

is greater than a given threshold $T$ where we used $T = 0.3$ throughout our experiments. The case counts as a true positive ($tp_i = 1$, $fp_j = 0$) for our evaluation. If there does not exist any $j$ that exceeds the IoU threshold $T$, a false negative ($fn_i = 1$, $fp_j = 0$ otherwise) is registered. Vice versa, if there does not exist any $i$ of $\{P^y_i\}$, such that $\text{IoU}_{ij} \geq T$, then the polygon $P^\hat{y}_j$ is counted as a false positive ($fp_j = 1$, $fp_j = 0$ otherwise).

Fig. 6 demonstrates examples of all three situations. The procedure is repeated for all pairs of geo-referenced test data maps $y$ with corresponding generated map $\hat{y} = G_Y(x)$.

The true positive count of the data set $TP = \sum_{i,j} t p_i$ is the total number of true positives for all samples $y \in Y$. In the same manner, the false positive count is computed according to $FP = \sum_{i,j} f p_i$ from all $\hat{y} \in \{G_Y(x) : x \in X\}$. Finally, we have the false negative count determined by $FN = \sum_{i} fn_i$.

Once the integrated quantities $TP$, $FP$, and $FN$ are obtained, precision $p$ and recall $r$ is computed by their standard definitions:

$$p = \frac{TP}{TP + FP} \quad (4)$$
$$r = \frac{TP}{TP + FN} \quad (5)$$

In addition, the $F_1$-score that resembles the harmonic mean of precision and recall is defined through

$$f_1 = \frac{1}{\frac{1}{p} + \frac{1}{r}} = \frac{2pr}{p + r} \quad (6)$$

As already discussed, neither is the OSM labels complete in terms of houses, nor it is accurate in terms of roads for rural areas. By way of experiment, however, we found that most of the house labels are accurate, if existing. Therefore, we assume house labels to be incomplete, but accurate, and hence, we restrict ourself to the recall score $r$ as a measure to improve overall model performance for detecting houses. We provide a discussion on the precision score $p$ as complement in Sect. 7.4 employing human, visual inspection.

6 EXPERIMENTAL SETUP

Our experiments were developed and performed on a cluster of 4 servers. Each machine has 14-core Intel Xeon E5-2680 v4 2.40GHz processors, 1TB main memory, and 4 Nvidia P100 GPUs. GPUs and CPUs are connected via PCIe Gen3 bus with 16GB/s peak bandwidth in each direction. The servers are connected by 100Gbit/s ethernet.

PyTorch version 1.1.0 is the underlying deep learning framework in use. We use Nvidia’s CUDA 9.2 API model, the CUDA-aware OpenMPI v3.1.1, and the GNU C++ compiler version 4.8.5 to build our communication library, which connects with PyTorch via a Python-C interface.

Table 1: House density (average number of labeled houses per square kilometer) and completeness score for each dataset.

| City      | House Density | Completeness Score |
|-----------|---------------|--------------------|
| Vegas     | 3283          | 100%               |
| Austin    | 1723          | 52%                |
| Dallas    | 1285          | 39%                |
| San Antonio | 95            | 3%                 |
| Houston   | 141           | 4%                 |

Table 2: U-Net and FW-CycleGAN Comparison on SpaceNet Vegas Dataset.

| Model     | Train City | Test City | Precision | Recall | F1    |
|-----------|------------|-----------|-----------|--------|-------|
| U-Net     | Vegas      | Vegas     | 0.829     | 0.821  | 0.825 |
| CycleGAN  | Vegas      | Vegas     | 0.700     | 0.414  | 0.520 |

7 RESULTS AND DISCUSSION

As discussed above, in this study we focused on four cities in Texas, namely: Austin, Dallas, San Antonio, and Houston. After all data tiles had been exported from PAIRS, in a first step, we applied an entropy threshold on the tile’s pixel value distribution. It enabled us to filter out tiles dominated by bare land with few features such as houses and roads. Then, each collection of tiles is randomly split into training and testing with split ratio 4/1.

It is found that among the four cities in Texas extracted for this study, the house density\(^3\) varies significantly as summarized in Table 1. Given the four cities are located in the same state, one would expect the density of houses to be relatively similar, yet the number of house labels as provided by the OSM map varies by more than one order of magnitude. For our setting, we consider the SpaceNet dataset as most complete in terms of house labels. Although Las Vegas, NV is not in the state of Texas, lacking of a better alternative we used the its house density for book-keeping purpose to compute the completeness score. We define the completeness score by the ratio of the house density of any Texas city vs. the house density in Las Vegas. The house densities and corresponding completeness scores are listed in Table 1. House density is a critical variable for model performance, as a less complete dataset generates a more biased model, thus impacting overall accuracy. After we present our findings on model performance in view of data completeness below, we detail on how we incrementally fill missing data in order to improve overall model accuracy.

7.1 Model Comparison on Datasets with Different Level of Completeness

In a first step, we established a comparison of U-Net vs. CycleGAN using the most accurate and complete dataset from Las Vegas. Results are summarized in Table 2. As expected, the winning architecture in the SpaceNet building detection challenge performs much better than the CycleGAN model. We note that for the SpaceNet Las Vegas dataset, houses are the only labels, i.e. in the rasterized map used for training, no road labels exist. In our experiments we observed that CycleGAN is challenged by translating satellite images with rich features such as roads, parks, lots, etc. into void on the generated map. Thus the task is not necessarily suited for such an architecture.

\(^3\)defined as average number of houses labeled per square kilometer
After we did train a U-Net, FW-CycleGAN demonstrated, the model slightly outperformed the model trained on the SpaceNet Las Vegas data, which has a F1 score of 88.5% for Las Vegas, NV down to 25% in Austin, TX. Hence, the result underlines the need for a training dataset with wide variety of scenes across different geo-locations in order to be able to generate accurate maps.

Last, we compared the CycleGAN, the FW-CycleGAN and the U-Net models using the Austin dataset. Corresponding results are shown in Table 4. We observe a drop in recall from 82% in Las Vegas, NV down to 25% in Austin, TX. Hence, the result underlines the need for a training dataset with wide variety of scenes across different geo-locations in order to be able to generate accurate maps.

Table 3: The U-Net model generalization test cases.

| Model   | Train City | Test City | Precision | Recall | F1   |
|---------|------------|-----------|-----------|--------|------|
| U-Net   | Vegas      | Houston   | 0.829     | 0.370  | 0.507|
| U-Net   | Austin     | Houston   | 0.816     | 0.732  | 0.772|

Table 4: CycleGAN, FW-CycleGAN and U-Net Comparison on PAIRS dataset.

| Model     | Train City | Test City | Precision | Recall | F1    |
|-----------|------------|-----------|-----------|--------|-------|
| CycleGAN  | Austin     | Austin    | 0.546     | 0.664  | 0.618 |
| FW-CycleGAN | Austin    | Austin    | 0.641     | 0.740  | 0.687 |
| U-Net     | Austin     | Austin    | 0.816     | 0.732  | 0.772 |

Table 5: Generalization comparison between FW-CycleGAN and U-Net

| Model     | Train City | Test City | Precision | Recall | F1    |
|-----------|------------|-----------|-----------|--------|-------|
| FW-CycleGAN | Austin    | Austin    | 0.641     | 0.740  | 0.687 |
| U-Net     | Austin     | Austin    | 0.816     | 0.732  | 0.772 |
| U-Net     | Austin     | San Antonio | 0.032   | 0.499  | 0.059 |
| FW-CycleGAN | Austin    | Houston   | 0.040     | 0.470  | 0.074 |
| U-Net     | Austin     | Houston   | 0.045     | 0.370  | 0.080 |

7.2 Model comparison on generalization

In a next step, we wanted to investigate the generalization capability of the U-Net model trained on the accurate and complete SpaceNet data in Las Vegas. If the model would be able to generalize from one geo-location to another, the amount of data needed to train a model for the entire area of CONUS would be significantly reduced. After we did train a U-Net model on the SpaceNet Las Vegas data, inference was performed on the Austin, TX dataset from OSM. The results are summarized in Table 3. We observe a drop in recall from 82% in Las Vegas, NV down to 25% in Austin, TX. Hence, the result underlines the need for a training dataset with wide variety of scenes across different geo-locations in order to be able to generate accurate maps.

Last, we compared the CycleGAN, the FW-CycleGAN and the U-Net models using the Austin dataset. Corresponding results are shown in Table 4. We demonstrated that the additional FW loss significantly improved the recall of the CycleGAN increasing it to 74.0% from a baseline CycleGAN that yielded a value of 46.4%. Also, the FW-CycleGAN model slightly outperformed the U-Net which achieved a recall of 73.2%.

In yet another case study, we numerically determined the recall of the two best performing models, namely FW-CycleGAN and U-Net, on other cities. The results are summarized in Table 5. As demonstrated, the FW-CycleGAN model consistently generates better recall values across all the three cities in Texas other than Austin.

7.3 Incremental Data Augmentation

Given the assumption that maps of cities from the same state follow the same statistics regarding features, we propose a data augmentation scheme to incrementally fill in missing labels in less complete datasets. The incremental data augmentation scheme uses a model trained on a more completely labeled dataset, e.g., Austin area, to generate maps for a less complete datasets, e.g., geo-locations of Dallas and San Antonio.

Table 5: Generalization comparison between FW-CycleGAN and U-Net.

| Model     | Train City | Test City | Precision | Recall | F1    |
|-----------|------------|-----------|-----------|--------|-------|
| FW-CycleGAN | Austin    | Austin    | 0.641     | 0.740  | 0.687 |
| U-Net     | Austin     | Austin    | 0.816     | 0.732  | 0.772 |
| FW-CycleGAN | Austin    | San Antonio | 0.032   | 0.499  | 0.059 |
| FW-CycleGAN | Austin    | Houston   | 0.040     | 0.470  | 0.074 |
| U-Net     | Austin     | Houston   | 0.045     | 0.370  | 0.080 |

7.4 Discussion on Precision

The precision score evaluated using OSM labels as ground truth is negatively impacted by the incompleteness of the OSM data. Indeed, we observed that OSM labels frequently miss houses whose presence is evident from NAIP imagery. In order to provide a quantitative understanding on the true model performance, we took the Austin test dataset as a case study: We randomly picked 100 samples of the aerial images, and manually surveyed these images.
We used Decentralized Parallel SGD (DPSGD) [13] to accelerate FW-CycleGAN. This way, the weight updating and averaging step of the two sets placed in a communication ring. After every mini-batch update, the stochastic gradient descent algorithm renders simultaneously. An architecture that employs one network and one objective function renders the stepping function of training. We rearranged the weight update steps for both the generators and the discriminators such that the stepping function of training utilizing 16 GPUs. As a result, we achieved a speed-up of 14.7 utilizing 16 GPUs, hence reducing the training time of CycleGAN from roughly 122 hours on one GPU down to 8.28 hours employing the decentralized parallel training utilizing 16 GPUs.

### 8 PUBLIC ACCESS TO GENERATED MAPS

Given the big geo-spatial data platform PAIRS discussed in Sect. 3.1, we make available inferred maps of our models to the public. The open-source Python module ibmpairs can be downloaded from https://github.com/ibm/ibmpairs. In order to retrieve the generated map features as colored overlay along with geo-referenced NAIP RGB image in the background, the following JSON load:

```json
{
    "layers": [
        {
            "id": "50155",
            "aggregation": "Max",
            "temporal": {
                "intervals": [{
                    "start": "2014-1-1",
                    "end": "2016-1-1"
                }]
            }
        },
        {
            "id": "50136",
            "aggregation": "Max",
            "temporal": {
                "intervals": [{
                    "start": "2014-1-1",
                    "end": "2016-1-1"
                }]
            }
        },
        {
            "id": "50138",
            "aggregation": "Max",
            "temporal": {
                "intervals": [{
                    "start": "2014-1-1",
                    "end": "2016-1-1"
                }]
            }
        },
        {
            "id": "49240",
            "aggregation": "Max",
            "temporal": {
                "intervals": [{
                    "start": "2014-1-1",
                    "end": "2016-1-1"
                }]
            }
        },
        {
            "temporal": {
                "intervals": [{
                    "start": "2014-1-1",
                    "end": "2016-1-1"
                }]
            }
        }
    ],
    "coordinates": [32.6659568, -97.4756499, 32.6790701, -97.4465533],
    "type": "square"
}
```

can be submitted as query to PAIRS. The Python sub-module paw of ibmpairs utilizes the PAIRS’s core query RESTful API. Example code reads:

```python
# set up connection to PAIRS from ibmpairs import paw
import os
os.environ["PAW_PAIRS_DEFAULT_PASSWORD_FILE_NAME"] = "/path/to/pairs/password/file"

os.environ["PAW_PAIRS_DEFAULT_USER"] = "pairsuser@somedomain.org"
paw.load_environment_variables()
paw.retrieve_generated_map_and_associated_imagery_fromPAIRS
    query = paw.PAIRSQuery(queryJSON)
query.submit()
query.poll_till_finished()
query.download()
query.create_layers()
```

with queryJSON the JSON load listed above, assuming a fictitious user pairsuser@somedomain.org. Overall, six PAIRS raster layers are queried corresponding to 3 RGB channels for the satellite imagery and 3 RGB channels for the generated, rasterized map.

### 9 CONCLUSION AND FUTURE WORK

In this paper, we performed a case study to investigate the quality of publicly available data in the context of map generation from

---

Corresponding pip and conda package are available through https://pypi.org/project/ibmpairs/ and https://anaconda.org/conda-forge/ibmpairs, respectively; i.e running pip install ibmpairs or conda install -c conda-forge ibmpairs is a way to get the Python module.
high-resolution aerial/satellite imagery by applying deep learning architectures. In particular, we utilized the aerial images from the NAIP program to characterize rasterized maps based on the crowdsourced OSM dataset.

We confirmed that geographic, economic, and cultural heterogeneity renders significant differences on man-made infrastructure which calls for more broadly available training data like the ones used in this study. We employed two state-of-the-art deep convolution neural network models, namely U-Net and CycleGAN. Furthermore, based on the objective, we introduced the Feature-Weighted CycleGAN which significantly improved binary classification accuracy for house detection. Although OSM is not accurate in rural areas, and it is incomplete in urban areas, we assumed: once a house is labeled, it is accurately labeled. Consequently, we focused our model performance evaluation on recall. In addition, we provided manually obtained evaluation metrics, to show that both precision and recall increases in value in accurately labeled subsets.

For scenarios where the incompleteness of OSM labels is significant, we propose an incremental data augmentation scheme that has significantly improved model accuracy in such areas. Even for cities which are relatively complete in terms of labeling, the data augmentation scheme helped lifting the best recall to 84%, and our manual count of binary classification of the Austin dataset shows the precision score is above 84%, yielding a F1 score prominently close to a corresponding SpaceNet winning solution exploiting more data input compared to our approach.

Obviously, to finally map the entire world we need to deal with enormous amounts of training data. To this end, we applied an Decentralized Parallel Stochastic Gradient Descent (DP-SGD) training scheme that is scalable to hundreds of GPUs with near linear speed-up. At the same time it carries the same level of convergence as compared to non-parallel training schemes. We demonstrated an implementation of the DP-SGD scheme and achieved a speed-up of 14.7 times over a cluster of 16 GPUs. Nevertheless, we observed a gap in convergence. Tuning the code for further improvement is the subject of our current, ongoing agenda.

REFERENCES

[1] [n.d.]. OpenStreetMap. https://www.openstreetmap.org. Accessed: 2020-02-01.
[2] [n.d.]. SpaceNet – Accelerating Geospatial Machine Learning. https://spacenet.ai/. Accessed: 2020-02-03.
[3] Derrick Bonafilia, James Gill, Saikat Basu, and David Yang. 2019. Building High Resolution Maps for Humanitarian Aid and Development with Weakly-and Semi-Supervised Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 1–9.
[4] Open Geospatial Consortium. 2019. OGC GeoTIFF standard. http://docs.opengeospatial.org/is/19-008r4/19-008r4.html
[5] Ilke Demir, Krzysztof Koperski, David Lindenbaum, Guan Pang, Jing Huang, Saikat Basu, Forest Hughes, Devis Tuia, and Ramesh Raskar. 2018. DeepGlobe 2018: A Challenge to Puzzle the Earth Through Satellite Images. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops.
[6] Adam Van Etten, Dave Lindenbaum, and Todd M. Bacastow. 2018. SpaceNet: A Remote Sensing Dataset and Challenge Series. arXiv:cs.CV/1807.01232
[7] Raphael A. Finkel and Jon Louis Bentley. 1974. Quad trees a data structure for retrieval on composite keys. Acta informatica 4, 1 (1974), 1–9.
[8] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Koskela, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2017. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. arXiv:1706.02677
[9] Suyog Gupta, We Zhang, and Fei Wang. 2016. Model Accuracy and Runtime Tradeoff in Distributed Deep Learning: A Systematic Study. 171–180. https://doi.org/10.1109/TCDM.2016.0028
[10] IEEE. 2015. Towards a Definition of the Internet of Things (IoT). https://iot.ieee.org/definition.html
[11] Pascal Kaiser, Jan Dirk Wegner, Aurelien Lucchi, Martin Jaggi, Thomas Hofmann, and Konrad Schindler. 2017. Learning aerial image segmentation from online maps. IEEE Transactions on Geoscience and Remote Sensing 55, 11 (2017), 6654–6668.
[12] Levente J Klein, Fernando J Marianno, Conrad M Albrecht, Marcus Freitag, Siyuan Lu, Nigel Hinds, Xiaoyan Shao, Sergio Bermudez Rodriguez, and Hendrik F Hamann. 2015. PAIRS: A scalable geo-spatial data analytics platform. In 2015 IEEE International Conference on Big Data (Big Data). IEEE, 1290–1298.
[13] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. 2017. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. In Advances in Neural Information Processing Systems 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc.: 5338–5340. http://papers.nips.cc/paper/7117-can-decentralized-algorithms-outperform-centralized-algorithms-a-case-study-for-decentralized-parallel-stochastic-gradient-descent.pdf
[14] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. 2018. Asynchronous Decentralized Parallel Stochastic Gradient Descent. In Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018. 3049–3058. http://proceedings.mlr.press/v80/lian18a.html
[15] Siyuan Lu, Xiaoyan Shao, Marcus Freitag, Levente J Klein, Jason Renwick, Fernando J Marianno, Conrad Albrecht, and Hendrik F Hamann. 2016. IBM PAIRS curated big data service for accelerated geo-spatial data analytics and discovery. In 2016 IEEE International Conference on Big Data (Big Data). IEEE, 2672–2675.
[16] S. Lu, X. Shao, M. Freitag, L. J. Klein, J. Renwick, F. J. Marianno, C. Albrecht, and H. F. Hamann. 2016. IBM PAIRS curated big data service for accelerated geospatial data analytics and discovery. In 2016 IEEE International Conference on Big Data (Big Data). 2672–2675. https://doi.org/10.1109/BigData.2016.7840910
[17] Emmanuel Maggi, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez. 2016. Convolutional neural networks for large-scale remote-sensing image classification. IEEE Transactions on Geoscience and Remote Sensing 55, 2 (2016), 645–657.
[18] Mapnik. 2005. Mapnik. https://github.com/mapnik/mapnik/.
[19] Microsoft. 2018. US Building Footprints. https://github.com/microsoft/USBuildingFootprints.
[20] U.S. Department of Agriculture. 2020. National Agriculture Imagery Program. https://www.fsa.usda.gov/programs-and-services/airial-photography/imagery-program/naip-imagery/.
[21] Kohei Otaki. 2019. Winning Solution for the Spacenet Challenge: Joint Learning with OpenStreetMap https://hi.co/winning-solution-for-the-spacenet-challenge-joint-learning-with-opencities.html
[22] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. Medical Imaging Computing and Computer-Assisted Intervention AÂŠMICCAI (2015). 234â–245. https://doi.org/10.1007/978-3-319-25474-4_28
[23] Gaurav Sharma. 2003. Digital Color Imaging Handbook. CRC Press.
[24] Suriya Singh, Anil Batra, Guan Pang, Lorenzo Torrenesi, Saikat Basu, Manohar Paluri, and CV Jawahar. 2018. Self-Supervised Feature Learning for Semantic Segmentation of Overhead Imagery. In BMVC. Vol. 1. 4.
[25] Tobias G Tzec, Xianming Liu, Amy Zhang, Lin Li, Gregory Yetman, Taip Kile, Simon Murray, Brian Blankenspoor, Esper B Prydz, et al. 2017. Mapping the world population one building at a time. arXiv preprint arXiv:1712.05839 (2017).
[26] Mehul Nalin Vora. 2011. Hadoop-HBase for large-scale data. In Proceedings of the International Conference on Computer Science and Network Technology. Vol. 1. IEEE, 601–605.
[27] Wai Yeung Yeung, Dhananjay Jagtap, and Nagwa El-Ashmawy. 2015. Urban land cover classification using airborne LiDAR data: A review. Remote Sensing of Environment 158 (2015), 295–310
[28] Matei Zaharia, Reynold S Xin, Patrick Wendell, Tathagata Datta, Brian Blanke, Brian Blankenspoor, Esper B Prydz, and Joseph S GN. 2010. Increasing the speed of large-scale distributed programs. In SIGMOD, 255–266.
[29] Wei Zhang, Suyog Gupta, Xiangru Lian, and Ji Liu. 2016. Staleness-Aware Asynchronous Decentralized Parallel Stochastic Gradient Descent. In Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2016. 3049–3058. http://proceedings.mlr.press/v48/lian16a.html
[30] Wei Zhang, Suyog Gupta, Xiangru Lian, and Ji Liu. 2018. Asynchronous Decentralized Parallel Stochastic Gradient Descent. In Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018. 3049–3058. http://proceedings.mlr.press/v80/lian18a.html