City-Scale Road Extraction from Satellite Imagery v2: 
Road Speeds and Travel Times

Adam Van Etten
In-Q-Tel CosmiQ Works
avanetten@iqt.org

Abstract

Automated road network extraction from remote sensing imagery remains a significant challenge despite its importance in a broad array of applications. To this end, we explore road network extraction at scale with inference of semantic features of the graph, identifying speed limits and route travel times for each roadway. We call this approach City-Scale Road Extraction from Satellite Imagery v2 (CRESIv2). Including estimates for travel time permits true optimal routing, not just the shortest geographic distance. We compare SpaceNet labels to OpenStreetMap (OSM) labels, and find that models both trained and tested on SpaceNet labels outperform OSM labels by ≥60%. For a diverse test set of SpaceNet data and a traditional edge weight of geometric distance, we find an aggregate of 5% improvement over existing methods. We also test our algorithm on Google satellite imagery with OpenStreetMap labels, and find a 23% improvement over previous work. Metric scores decrease by only 4% on large graphs when using travel time rather than geometric distance for edge weights, indicating that optimizing routing for travel time is feasible with this approach.

1. Introduction

The automated extraction of roads applies to a multitude of long-term efforts: improving access to health services, urban planning, and improving social and economic welfare. This is particularly true in developing countries that have limited resources for manually intensive labeling and are under-represented in current mapping. Updated maps are also crucial for such time-sensitive efforts as determining communities in greatest need of aid, effective positioning of logistics hubs, evacuation planning, and rapid response to acute crises.

Existing data collection methods such as manual road labeling or aggregation of mobile GPS tracks are currently insufficient to properly capture either underserved regions (due to infrequent data collection), or the dynamic changes inherent to road networks in rapidly changing environments. For example, in many regions of the world OpenStreetMap (OSM) road networks are remarkably complete. Yet, in developing nations OSM labels are often missing metadata tags (such as speed limit or number of lanes), or are poorly registered with overhead imagery (i.e., labels are offset from the coordinate system of the imagery), see Figure 1. An active community works hark to keep the road network up to date, but such tasks can be challenging and time consuming in the face of large scale disasters. For example, following Hurricane Maria, it took the Humanitarian OpenStreetMap Team (HOT) over two months to fully map Puerto Rico [20].

The frequent revisits of satellite imaging constellations may accelerate existing efforts to quickly update road network and routing information. A fully automated approach to road network extraction and travel time estimation from satellite imagery therefore warrants investigation, and is explored in the following sections. In Section 2, we discuss related work, while Section 3 details the graph extraction algorithm that infers a road network with semantic features directly from imagery. In Section 4, we discuss our dataset and our method for assigning road speed estimates based on road geometry and metadata tags. Section 5 discusses the...
need for modified metrics to measure our semantic graph, and Section 6 covers our experiments to extract road networks from multiple datasets. Finally in Sections 7 and 8 we discuss our findings and conclusions, respectively.

2. Related Work

Extracting road pixels in small image chips from aerial imagery has a rich history (e.g. [33], [19], [29], [33], [25], [15]). These algorithms generally use a segmentation + post-processing approach combined with lower resolution imagery (resolution ≥ 1 meter), and OpenStreetMap labels.

Extracting road networks directly has also been attempted by a number of studies. [26] attempted road extraction via a Gibbs point process, while [31] showed some success with road network extraction with a conditional random field model. [6] used junction-point processes to recover line networks in both roads and retinal images [27] extracted road networks by representing image data as a graph of potential paths. [18] extracted road centerlines and widths via OSM and a Markov random field process. [16] used a topology-aware loss function to extract road networks from aerial features as well as cell membranes in microscopy.

Of greatest interest for this work are a trio of recent papers that extracted road networks from overhead imagery. [17] used a segmentation followed by $A^*$ search, applied to the not-yet-released Toronto City Dataset. The RoadTracer paper utilized an interesting approach that used OSM labels to directly extract road networks from imagery without intermediate steps such as segmentation. While this approach is compelling, according to the authors it “struggled in areas where roads were close together” [2] and underperforms other techniques such as segmentation+post-processing when applied to higher resolution data with dense labels. [3] used a connectivity task called they termed Orientation Learning, combined with a stacked - convolutional module to effectively utilize the mutual information between orientation learning and segmentation tasks to extract road networks from satellite imagery, noting improved performance over [3] on small image chips.

We build upon CRESI v1 [9] that scaled up narrow-field road network extraction methods; in this work we focus primarily on developing methodologies to infer road speeds and travel times, but also improve algorithmic performance and inference speed.

3. Road Network Extraction Algorithm

Our approach is to combine novel segmentation approaches that encapsulate road speed, with improved post-processing techniques for road vector simplification. We utilize satellite imagery and geocoded road centerline labels (see Section 4 for details on data) to build training datasets for the first step in our algorithm: segmentation models.

We create training masks from road centerline labels, assuming a mask halfwidth of 2 meters for each edge. While scaling the training mask width with the full width of the road is an option (e.g. a four lane road will have a greater width than a two lane road), since the end goal is road centerline vector extraction, we utilize the same training mask width for all roadways.

We have two goals: extract the road network over large areas, and assess travel time along each roadway. In order to assess travel time we assign a speed limit to each roadway based on metadata tags such as road type, number of lanes, and surface construction. We subsequently train models according to the segmentation approaches detailed below.

3.1. Multi-Class Segmentation

We create multi-channel training masks by binning the road labels into a 7-layer stack, with channel 0 detailing speeds between 1-10 mph, channel 1 between 11-20 mph, etc. (see Figure 2). We train a segmentation model inspired by the winning SpaceNet 3 algorithm [1], and use a ResNet34 encoder with a U-Net inspired decoder. We include skip connections every layer of the network, with an Adam optimizer and a custom loss function of:

$$\mathcal{L} = \alpha_{mc} \mathcal{L}_\text{focal} + (1-\alpha) \mathcal{L}_\text{dice}$$

where ‘focal’ is focal loss [13], ‘dice’ is the Dice coefficient, and $\alpha_{mc}$ is a constant.

3.2. Continuous Mask Segmentation

A second segmentation method renders continuous training masks from the road speed labels. Rather than the typical binary mask, we scale the mask value with speed limit, with a maximum burn value of 255 (for 65 mph), and a minimum burn value of 40 for the rare 15 mph roads (see Figure 2).

We use a similar network architecture to the previous section (ResNet34 encoder with a U-Net inspired decoder), though we use a loss function of:

$$\mathcal{L} = \alpha_c \mathcal{L}_\text{CE} + (1-\alpha) \mathcal{L}_\text{dice}$$

where ‘CE’ is cross entropy, ‘dice’ is the Dice coefficient, and $\alpha_c$ is a constant.

3.3. Graph Extraction Procedure

The output of the segmentation mask step detailed above is subsequently refined into road vectors. We begin by smoothing and flattening the final output mask to create a binary prediction mask. This binary mask is then refined using techniques such as opening and closing, from which we create a skeleton (e.g. scikit-image skeletonize [24]). This skeleton is rendered into a graph structure with a modified
Figure 2: Training data. (a) Input image. (b) Typical binary road training mask (not used in this study). (c) Multi-class training mask showing individual speed channels: red = 21-30 mph, green = 31-40 mph, blue = 41-50 mph. (d) Continuous training mask, whiter denotes higher speeds.

Figure 3: Graph extraction procedure. Left: merged mask output. Left center: refined mask. Right center: mask skeleton. Right: graph structure.

A version of the *sknw* package [32]. This process is detailed in Figure 3. The graph created by this process contains length information for each edge, but no other metadata. To close small gaps and remove spurious connections not already corrected by the opening and closing procedures, we remove disconnected subgraphs with an integrated path length of less than a certain length. We also follow [1] and remove terminal vertices that lie on an edge less than 10 pixels in length, and connect terminal vertices if the distance to the nearest non-connected node is less than 20 pixels.

3.4. Speed Estimation Procedure

We estimate travel time for a given road edge by leveraging the speed information encapsulated in the prediction mask. The majority of edges in the graph are composed of multiple segments (e.g. the edge in Figure 4 has 6 segments). Accordingly, we attempt to estimate the speed of each segment in order to determine the mean speed of the edge. This is accomplished by analyzing the prediction mask at the location of the segment midpoints. For each segment in the edge, at the location of the midpoint of the segment we extract a small $8 \times 8$ pixel patch from the prediction mask. The speed of the patch is estimated by filtering out low probability values (likely background), and averaging the remaining pixels. In the multi-class case, if the majority of the high confidence pixels in the prediction mask patch belong to channel 3 (corresponding to 31-40 mph), we would assign the speed at that patch to be 35 mph. For the continuous case the inferred speed is simply directly proportional to the mean pixel value.

The travel time for each edge is in theory the path integral of the speed estimates at each location along the edge. But given that each roadway edge is presumed to have a constant speed limit, we refrain from computing the path integral along the graph edge. Instead, we estimate the speed limit of the entire edge by taking the mean of the speeds at each segment midpoint. Travel time is then calculated as edge distance divided by mean speed.
Large image segmentation. BASISS process of slicing a large satellite image (top) and ground truth road mask (bottom) into smaller cutouts for segmentation training or inference [7].

Table 1: CRESIv2 Inference Algorithm

| Step | Description |
|------|-------------|
| 1    | Split large test image into smaller windows |
| 2    | Apply multi-class segmentation model to each window |
| 2b   | * Apply remaining (3) cross-validation models |
| 2c   | * For each window, merge the 4 predictions |
| 3    | Stitch together the total normalized road mask |
| 4    | Clean road mask with opening, closing, smoothing |
| 5    | Skeletonize flattened road mask |
| 6    | Extract graph from skeleton |
| 7    | Remove spurious edges and close small gaps in graph |
| 8    | Estimate local speed limit at midpoint of each segment |
| 9    | Assign travel time to each edge from aggregate speed |

* Optional

3.5. Scaling to Large Images

The process detailed above works well for small input images, yet fails for large images due to a saturation of GPU memory. For example, even for a relatively simple architecture such as U-Net [23], typical GPU hardware (NVIDIA Titan X GPU with 12 GB memory) will saturate for images greater than \(~2000 \times 2000\) pixels in extent and reasonable batch sizes > 4. In this section we describe a straightforward methodology for scaling up the algorithm to larger images. The first step in this methodology is provided by the Broad Area Satellite Imagery Semantic Segmentation (BASISS) [7] methodology; this approach is outlined in Figure 5 and returns a road pixel mask for a large test image.

The final algorithm is given by Table 1. The output of the CRESIv2 algorithm is a NetworkX [10] graph structure, with full access to the many algorithms included in this package.

4. Data

Many existing publicly available labeled overhead or satellite imagery datasets tend to be relatively small, or labeled with lower fidelity than desired for foundational mapping. For example, the International Society for Photogrammetry and Remote Sensing (ISPRS) semantic labeling benchmark [12] dataset contains high quality 2D semantic labels over two cities in Germany and covers a compact area of 4.8 km²; imagery is obtained via an aerial platform and is 3 or 4 channel and 5-10 cm in resolution. The TorontoCity Dataset [30] contains high resolution 5-10 cm aerial 4-channel imagery, and \(~700\) km² of coverage; building and roads are labeled at high fidelity (among other items), but the data has yet to be publicly released. The Massachusetts Roads Dataset [14] contains 3-channel imagery at 1 meter resolution, and 2600 km² of coverage; the imagery and labels are publicly available, though labels are scraped from OpenStreetMap and not independently collected or validated. The large dataset size, higher resolution (0.3 meter resolution), and hand-labeled and quality controlled labels of SpaceNet [28] provide an opportunity for algorithm improvement. In addition to road centerlines, the SpaceNet dataset contains metadata tags for each roadway including: number of lanes, road type (e.g. motorway, residential, etc), road surface type (paved, unpaved), and bridgeway (true/false).

4.1. SpaceNet Data

Our primary dataset accordingly consists of the SpaceNet 3 DigitalGlobe satellite imagery (30 cm/pixel) and attendant road centerline labels. Imagery covers 3000 square kilometers, and over 8000 km of roads are labeled [28]. Training images and labels are tiled into \(1300 \times 1300\) pixel \((\approx 160,000\, \text{m}^2)\) chips (see Figure 6).

We assign a maximum safe traversal speed of 10 - 65 mph to each segment based on the road metadata tags. For example, a paved one-lane residential road has a speed limit of 25 mph, a three-lane paved motorway can be traversed at 65 mph, while a one-lane dirt cart track has a traversal speed of 10 mph.

Figure 6: SpaceNet training chip. Left: SpaceNet GeoJSON road label. Right: \(400 \times 400\) meter image overlaid with road centerline labels (orange).
speed of 15 mph. See Appendix A for further details. This approach is tailored to disaster response scenarios, where safe navigation speeds likely supersede government-defined speed limits. We therefore prefer estimates based on road metadata over government-defined speed limits, which may be unavailable or inconsistent in many areas.

To test the city-scale nature of our algorithm, we extract large test images from all four of the SpaceNet cities with road labels: Las Vegas, Khartoum, Paris, and Shanghai. As the labeled SpaceNet test regions are non-contiguous and irregularly shaped, we define rectangular subregions of the images where labels do exist within the entirety of the region. These test regions total 608 km², with a total road length of 9065 km, see Appendix B for further details.

4.2. Google / OSM Dataset

We also evaluate performance with the satellite imagery corpus used by [3]. This dataset consists of Google satellite imagery at 60 cm/pixel over 40 cities, 25 for training and 15 for testing. Vector labels are scraped from OSM, and we use these labels to build training masks according the procedures described above. Due to the high variability in OSM road metadata density and quality, we refrain from inferring road speed from this dataset, and instead leave this for future work.

5. Evaluation Metrics

Historically, pixel-based metrics (such as F1 score) have been used to assess the quality of road proposals, though such metrics are suboptimal for a number of reasons (see [28] for further discussion). Accordingly, we use the graph-theoretic Average Path Length Similarity (APLS) and map topology (TOPO) [5] metrics that were created to measure similarity between ground truth and proposal road graphs.

5.1. APLS Metric

To measure the difference between ground truth and proposal graphs, the APLS [28] metric sums the differences in optimal path lengths between nodes in the ground truth graph G and the proposal graph G’, with missing paths in the graph assigned a score of 0. The APLS metric scales from 0 (poor) to 1 (perfect). Missing nodes of high centrality will be penalized much more heavily by the APLS metric than missing nodes of low centrality. The definition of shortest path can be user defined; the natural first step is to consider geographic distance as the measure of path length (APLS$_{\text{length}}$), but any edge weights can be selected. Therefore, if we assign a travel time estimate to each graph edge we can use the APLS$_{\text{time}}$ metric to measure differences in travel times between ground truth and proposal graphs.

For large area testing, evaluation takes place with the APLS metric adapted for large images: no midpoints along edges and a maximum of 500 random control nodes.

5.2. TOPO Metric

The TOPO metric [5] is an alternative metric for computing road graph similarity. TOPO compares the nodes that can be reached within a small local vicinity of a number of seed nodes, categorizing proposal nodes as true positives, false positives, or false negatives depending on whether they fall within a buffer region (referred to as the “hole size”). By design, this metric evaluates local subgraphs in a small subregion (~ 300 meters in extent), and relies upon physical geometry. Connections between greatly disparate points (> 300 meters apart) are not measured, and the reliance upon physical geometry means that travel time estimates cannot be compared.

6. Experiments

We train CRESIv2 models on both the SpaceNet and Google/OSM datasets. For the SpaceNet models, we use the 2780 images/labels in the SpaceNet 3 training dataset. The Google/OSM models are trained with the 25 training cities in [3]. All segmentation models assign a road centerline halfwidth of 2 meters, and withhold 25% of the training data for validation purposes. Training occurs for 30 epochs. Optionally, one can create an ensemble of 4 folds (i.e. the 4 possible unique combinations of 75% train and 25% validate) to train 4 different models. This approach may increase model robustness, at the cost of increased compute time. As inference speed is one of our focii, all results shown below (save Section 6.3) use a single model, rather than the ensemble approach.

For the Google / OSM data, we train segmentation models as in Section 6.1, though with only a single class since we forego speed estimates with this dataset.

6.1. SpaceNet Test Corpus Results

We compute both APLS and TOPO performance for the 400 × 400 meter image chips in the SpaceNet test corpus. We utilize an APLS buffer and TOPO hole size of 4 meters (implying proposal road centerlines must be within 4 meters of ground truth). An example result is displayed in Figure 7 and Table 2. Reported errors ($\pm \sigma$) reflect the relatively high variance of performance among the various test scenes in the four SpaceNet cities. Table 2 indicates that the continuous mask model struggles to accurately reproduce road speeds, due in part to the models propensity to yield high mask pixel values for high confidence regions, thereby skewing speed estimates. In the remainder of the paper, we only consider the multi-class model. Table 2 also demonstrates that for the multi-class model the APLS score is still 0.58 when using travel time as the weight, which is only 13% lower than when weighting with geometric distance.
Table 2: Performance on SpaceNet Test Chips

| Model       | TOPO   | APLS<sub>length</sub> | APLS<sub>time</sub> |
|-------------|--------|-----------------------|---------------------|
| Multi-Class | 0.53 ± 0.23 | 0.68 ± 0.21       | 0.58 ± 0.21        |
| Continuous  | 0.52 ± 0.25 | 0.68 ± 0.22       | 0.39 ± 0.18        |

Figure 7: Algorithm performance on SpaceNet. (a) Ground truth and (b) predicted multi-class masks: red = 21-30 mph, green = 31-40 mph, blue = 41-50 mph, yellow = 51-60 mph. (c) Ground truth and (d) predicted graphs overlaid on the SpaceNet test chip; edge widths are proportional to speed limit. The scores for this proposal are APLS<sub>length</sub> = 0.80 and APLS<sub>time</sub> = 0.64.

6.2. Comparison of SpaceNet to OSM

As a means of comparison between OSM and SpaceNet labels, we use our baseline algorithm to train two models on SpaceNet imagery. One model uses ground truth masks rendered from OSM labels, while the other model uses the exact same algorithm, but uses ground truth segmentation masks rendered from SpaceNet labels. Table 3 displays APLS scores computed over a subset of the SpaceNet test chips, and demonstrates that the model trained and tested on SpaceNet labels is far superior to other combinations, with a ≈ 60 – 100% improvement. This is possibly due in part to the more uniform labeling schema and validation procedures adopted by the SpaceNet labeling team. The poor performance of the SpaceNet-trained OSM-tested model is likely due to a combination of different labeling density between the two datasets, and differing projections of labels onto imagery for SpaceNet and OSM data. Figure 8 illustrates the difference between predictions returned by the OSM and SpaceNet models.

Figure 8: SpaceNet compared to OSM. Road predictions (yellow) and ground truth SpaceNet labels (blue) for a sample Las Vegas image chip. OSM model predictions (a) are slightly more offset from ground truth labels than SpaceNet model predictions (b).

6.3. Ablation Study

In order to assess the relative importance of various improvements to our baseline algorithm, we perform ablation studies on the final algorithm. For evaluation purposes we utilize the the same subset of test chips as in Section 6.2 and the APLS<sub>length</sub> metric. Table 4 demonstrates that advanced post-processing significantly improves scores. Using a more complex network also improves the final prediction. Applying four folds improves scores very slightly, though at the cost of significantly increased algorithm runtime. Given the minimal improvement afforded by the ensemble step, all results in this paper (save this section) use only a single model.

Table 4: Road Network Ablation Study

| Description                                                     | APLS |
|----------------------------------------------------------------|------|
| 1 Extract graph directly from simple U-Net mask                | 0.56 |
| 2 Apply opening, closing, smoothing processes                  | 0.66 |
| 3 Close larger gaps using edge direction and length             | 0.72 |
| 4 Use ResNet34 + U-Net architecture                            | 0.77 |
| 5 Use 4 fold ensemble                                          | 0.78 |
6.4. Large Area SpaceNet Results

We apply the CRESIv2 algorithm described in Table 1 to the large area SpaceNet test set covering 608 km$^2$. Evaluation takes place with the APLS metric adapted for large images (no midpoints along edges and a maximum of 500 random control nodes), along with the TOPO metric. We use a buffer size (for APLS) or hole size (for TOPO) of 4 meters. We report scores in Table 5 as the mean and standard deviation of the test regions of in each city. Table 5 reveals an overall $\approx 4\%$ decrease in APLS score when using speed versus length as edge weights. This is somewhat improved from the decrease of 13% noted in Table 2, due primarily to the fewer edge effects from larger testing regions. Figure 9 displays the graph output of the algorithm typical urban environments, see Appendix C for further examples.

### Table 5: SpaceNet Large Area Performance

| Test Region | TOPO     | APLS$_{length}$ | APLS$_{time}$ |
|-------------|----------|-----------------|---------------|
| Khartoum    | 0.53 ± 0.09 | 0.64 ± 0.10    | 0.61 ± 0.05   |
| Las Vegas   | 0.63 ± 0.02 | 0.81 ± 0.04    | 0.79 ± 0.02   |
| Paris       | 0.43 ± 0.01 | 0.66 ± 0.04    | 0.65 ± 0.02   |
| Shanghai    | 0.45 ± 0.03 | 0.55 ± 0.13    | 0.51 ± 0.11   |
| Total       | 0.51 ± 0.02 | 0.67 ± 0.04    | 0.64 ± 0.03   |

6.5. Google / OSM Results

Our method achieves reasonable results on 60 cm data with OSM labels, as displayed in Figure 10. For the same 4 m APLS buffer used above, we achieve a score of APLS$_{length} = 0.53 \pm 0.11$. This score is below the SpaceNet results of Table 5 due to the lower resolution imagery and use of OSM labels, yet this score is consistent with the OSM - OSM results of Table 3.

6.6. Comparison to Previous Work

Table 6 demonstrates that CRESIv2 improves upon existing methods for road extraction, both on the 400 × 400 m SpaceNet image chips at 30 cm resolution. We also see an improvement on large 60 cm Google satellite images with OSM labels. To allow a direct comparison, we report the TOPO metric for CRESIv2 with the 15 m hole size used in [3]; a qualitative comparison is shown in Figure 11 (see Appendix D for more examples).

7. Discussion

CRESIv2 achieves superior performance to previous methods in extracting roads topology from satellite imagery, with a 5% improvement over previous efforts on the SpaceNet dataset, and a 23% improvement over previous efforts with Google satellite imagery + OSM labels. Routing based on time shows only a $3 - 13\%$ decrease from

---

**Figure 9:** CRESIv2 road speed. Output of CRESIv2 inference as applied to a portion of the SpaceNet Las Vegas large area test region. The APLS$_{length}$ score for this prediction is 0.85, and the APLS$_{time}$ score is 0.82. Roads are colored by inferred speed limit, from yellow (20 mph) to red (65 mph).

**Figure 10:** Inference on 60 cm imagery. Prediction for the Boston test region of the Google / OSM dataset. The APLS$_{length}$ score for this region is 0.53.
Table 6: Performance Comparison

| Algorithm               | Google / OSM (TOPO) | SpaceNet (APLS_{length}) |
|-------------------------|---------------------|--------------------------|
| DeepRoadMapper [17]     | 0.37                | 0.51 \( ^1 \)           |
| RoadTracer [3]          | 0.43                | 0.58 \( ^1 \)           |
| OrientationLearning [4] | -                   | 0.64                     |
| CRESIv2 (Ours)          | 0.53                | 0.67                     |

\( ^1 \) from Table 4 of [4]

Figure 11: **RoadTracer / CRESIv2.** Visual comparison between RoadTracer (left column, OSM labels in gray, predictions in yellow [8]) and CRESIv2 (right column, predictions in yellow) for various cities. From top: New York, Kansas City.

distance-based routing, indicating that true optimized routing is possible with this approach.

As with most approaches to automated road vector extraction, complex intersections are a challenge with CRESIv2. While we attempt to connect small gaps based on road direction, overpasses and onramps remain difficult to connect correctly (see Figure 12).

7.1. Inference Speed

Inference code has not been optimized for speed, but even so inference runs at a rate of 280 km\(^2\)/hour on a machine with a single Titan X GPU. At this speed, a 4-GPU cluster could map the entire 9100 km\(^2\) area of Puerto Rico in \( \approx \) 8 hours, a significant improvement over the two months required by human labelers [20].

7.2. CRESIv2 Challenges

Figure 12: **CRESIv2 challenges.** Zoom of CRESIv2 inference over an intersection in Shanghai. While the pixel-bases score of this prediction is very high, correctly connecting roadways in complex intersections remains elusive.

8. Conclusion

Optimized routing is crucial to a number of challenges, from humanitarian to military. Satellite imagery may aid greatly in determining efficient routes, particularly in cases involving natural disasters or other dynamic events where the high revisit rate of satellites may be able to provide updates far more quickly than terrestrial methods.

In this paper we demonstrated methods to extract city-scale road networks directly from remote sensing imagery. GPU memory limitations constrain segmentation algorithms to inspect images of size \( \sim 2000 \times 2000 \) pixels in extent, yet any eventual application of road inference must be able to process images far larger than a mere \( \sim \frac{1}{3} \) km\(^2\). Accordingly, we demonstrate methods to infer road networks and travel times for input images of arbitrary size. This is accomplished via a multi-step algorithm that segments small image chips, extracts a graph skeleton, refines nodes and edges, stitches chipped predictions together, extracts the underlying road network graph structure, and assigns speed limit / travel time properties to each roadway.

Measuring performance with the APLS graph theoretic metric we observe superior performance for models trained and tested on SpaceNet data over OSM data. When using OSM data, our method provides a significant (+23%) improvement over existing methods. Over a diverse test set that includes atypical lighting conditions, off-nadir observation angles, and locales with a multitude of dirt roads, we achieve a total score of \( \text{APLS}_{\text{length}} = 0.67 \), and nearly equivalent performance when optimizing for travel time: \( \text{APLS}_{\text{time}} = 0.64 \). Prediction is reasonably quick, with an inference speed of \( \geq 280 \) km\(^2\)/hour on a single GPU. While automated road network extraction is by no means a solved problem, the CRESIv2 algorithm improves upon and extends existing methods, with potential benefits to applications such as disaster response where rapid map updates are critical to success.

Acknowledgments We thank Nick Weir, Jake Shermeyer, and Ryan Lewis for their insight and feedback.
References

[1] albu. Spacenet round 3 winner: albu’s implementation. https://github.com/SpaceNetChallenge/RoadDetector/tree/master/albu-solution, 2018.

[2] F. Bastani. fbastani spacenet solution. https://github.com/FBastani/spacenet_solution, 2018.

[3] F. Bastani, S. He, S. Abbar, M. Alizadeh, H. Balakrishnan, S. Chawla, D. J. DeWitt, and S. Madden. Unthue: An incremental graph construction process for robust road map extraction from aerial images. CoRR, abs/1802.03680, 2018.

[4] A. Batra, S. Singh, G. Pang, S. Basu, C. Jawahar, and M. Paluri. Improved road connectivity by joint learning of orientation and segmentation. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

[5] J. Biagioni and J. Eriksson. Inferring road maps from global positioning system traces: Survey and comparative evaluation. Transportation Research Record, 2291(1):61–71, 2012.

[6] D. Chai, W. Frstner, and F. Lafarge. Recovering line-networks in images by junction-point processes. In 2013 IEEE Conference on Computer Vision and Pattern Recognition, pages 1894–1901, June 2013.

[7] CosmiQWorks. Broad area satellite imagery semantic segmentation. https://github.com/CosmiQ/basiiss, 2018.

[8] M. CSAIL. Roadtracer: Automatic extraction of road networks from aerial images. https://roadmaps.csail.mit.edu/roadtracer/, 2018.

[9] A. V. Etten. City-scale road extraction from satellite imagery. CoRR, abs/1904.09901, 2019.

[10] A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using networkx. In G. Varoquaux, T. Vaught, and J. Millman, editors, Proceedings of the 7th Python in Science Conference, pages 11–15, Pasadena, CA USA, 2008.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. CoRR, abs/1512.03385, 2015.

[12] ISPRS. 2d semantic labeling contest. http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html, 2018.

[13] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. CoRR, abs/1510.00121, 2016.

[14] V. Mnih. Machine Learning for Aerial Image Labeling. PhD thesis, University of Toronto, 2013.

[15] V. Mnih and G. E. Hinton. Learning to detect roads in high-resolution aerial images. In K. Daniilidis, P. Maragos, and N. Paragios, editors, Computer Vision – ECCV 2010, pages 210–223, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[16] A. Mosinska, P. Marquez-Neila, M. Kozinski, and P. Fua. Beyond the pixel-wise loss for topology-aware delineation. pages 3136–3145, 06 2018.

[17] G. Mityus, W. Luo, and R. Urtasun. Deeproadmapper: Extracting road topology from aerial images. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 3458–3466, Oct 2017.

[18] G. Mityus, S. Wang, S. Fidler, and R. Urtasun. Enhancing road maps by parsing aerial images around the world. In 2015 IEEE International Conference on Computer Vision (ICCV), pages 1689–1697, Dec 2015.

[19] G. Mityus, S. Wang, S. Fidler, and R. Urtasun. Hd maps: Fine-grained road segmentation by parsing ground and aerial images. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3611–3619, June 2016.

[20] OpenStreetMap. 2017 hurricanes irma and maria. https://wiki.openstreetmap.org/wiki/2017_Hurricanes_Irma_and_Maria, 2017.

[21] OpenStreetMap. Osm tags for routing/maxspeed. https://wiki.openstreetmap.org/wiki/OSM_tags_for_routing/Maxspeed#United_States_of_America, 01 2019.

[22] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org. https://www.openstreetmap.org, 2019.

[23] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. CoRR, abs/1505.04597, 2015.

[24] scikit image. Skeletonize. https://scikit-image.org/docs/dev/auto_examples/edges/plot_skeleton.html, 2018.

[25] A. Sironi, V. Lepetit, and P. Fua. Multiscale centerline detection by learning a scale-space distance transform. In 2014 IEEE Conference on Computer Vision and Pattern Recognition, pages 2697–2704, June 2014.

[26] R. Stoica, X. Descombes, and J. Zerubia. A gibbs point process for road extraction from remotely sensed images. International Journal of Computer Vision, 57:121–136, 05 2004.

[27] E. Tretken, F. Benmansour, B. Andres, H. Pfister, and P. Fua. Reconstructing loopy curvilinear structures using integer programming. In 2013 IEEE Conference on Computer Vision and Pattern Recognition, pages 1822–1829, June 2013.

[28] A. Van Etten, D. Lindenbaum, and T. M. Bacastow. SpaceNet: A Remote Sensing Dataset and Challenge Series. ArXiv e-prints, July 2018.

[29] J. Wang, Q. Qin, Z. Gao, J. Zhao, and X. Ye. A new approach to urban road extraction using high-resolution aerial image. ISPRS International Journal of Geo-Information, 5(7), 2016.

[30] S. Wang, M. Bai, G. Máttyus, H. Chu, W. Luo, B. Yang, J. Liang, J. Cheverie, S. Fidler, and R. Urtasun. Torontocity: Seeing the world with a million eyes. CoRR, abs/1612.00423, 2016.

[31] J. D. Wegner, J. A. Montoya-Zegarra, and K. Schindler. A higher-order crf model for road network extraction. In 2013 IEEE Conference on Computer Vision and Pattern Recognition, pages 1698–1705, June 2013.

[32] yxdragon. Skeleton network. https://github.com/yxdragon/sknw, 2018.

[33] Z. Zhang, Q. Liu, and Y. Wang. Road extraction by deep residual u-net. CoRR, abs/1711.0684, 2017.
Appendix A. Road Speed Assignment

See [28] for details on the precise labeling guidelines and road type definitions. We utilize road type (motorway, primary, secondary, tertiary, residential, unclassified, cart track) and road surface type (paved, non-paved) to assign speed to each edge.

Speed is assigned with Table [7] using the Oregon guidelines for road speed [21].

Table 7: Road Speeds (mph)

| Road Type     | 1 Lane | 2 Lane | 3+ Lane |
|---------------|--------|--------|---------|
| Motorway      | 55     | 55     | 65      |
| Primary       | 45     | 45     | 55      |
| Secondary     | 35     | 35     | 45      |
| Tertiary      | 30     | 30     | 35      |
| Residential   | 25     | 25     | 30      |
| Unclassified  | 20     | 20     | 20      |
| Cart Track    | 20     | 20     | 20      |

For each non-paved roadway, the speed from Table [7] is multiplied by 0.75 to give the final speed.

Appendix B. Large Area Test Data

Details of the large area testing regions for SpaceNet data are displayed in Table [8].

Table 8: Test Regions

| Test Region    | Area (Km²) | Road Length (Total Km) |
|----------------|------------|------------------------|
| Khartoum_0     | 3.0        | 76.7                   |
| Khartoum_1     | 8.0        | 172.6                  |
| Khartoum_2     | 8.3        | 128.9                  |
| Khartoum_3     | 9.0        | 144.4                  |
| Las_Vegas_0    | 68.1       | 1023.9                 |
| Las_Vegas_1    | 177.0      | 2832.8                 |
| Las_Vegas_2    | 106.7      | 1612.1                 |
| Paris_0        | 15.8       | 179.9                  |
| Paris_1        | 7.5        | 65.4                   |
| Paris_2        | 2.2        | 25.9                   |
| Shanghai_0     | 54.6       | 922.1                  |
| Shanghai_1     | 89.8       | 1216.4                 |
| Shanghai_2     | 87.5       | 663.7                  |
| Total          | 608.0      | 9064.8                 |

Figure 13: SpaceNet road vector labels over Shanghai (purple). The label boundary is discontinuous and irregularly shaped, so we define rectangular regions for testing purposes (e.g. the blue region denotes test region Shanghai_0).
Appendix C. CRESIv2 Road Speed Plots

Figure 14: **CRESIv2 road speed.** Output of CRESIv2 inference as applied to large SpaceNet test regions (from top: Khartoum, Paris, Shanghai). Roads are colored by inferred speed limit, from yellow (20 mph) to red (65 mph).

Appendix D. CRESIv2 / RoadTracer Visual Comparison

Figure 15: **RoadTracer / CRESIv2.** Performance comparison between RoadTracer (left column, OSM labels in gray, predictions in yellow [8]) and CRESIv2 (right column, predictions in yellow) for various cities. From top: Denver, Pittsburgh, Vancouver.