Combining Satellite Imagery and GPS Data for Road Extraction

Tao Sun, Zonglin Di and Yin Wang*
Tongji University, Shanghai
2nd ACM SIGSPATIAL International Workshop on AI for Geographic Knowledge Discovery (GeoAI 2018)
Outline

• Task Description

• Previous Solutions

• Proposed Method
  • The dataset we used

• Training Details
Task Description

- Automatically extracting roadmap from satellite images.
- The manual methods are costly, error-prone, and easily become outdated.
- It is a binary classification or segmentation problem.
Previous Solutions

- GPS: Kernel Density Estimation (KDE), …

- Satellite Images & CNN-based methods:
  - Classification model: AlexNet, VGG, ResNet, …
  - Segmentation model: Fully Convolutional Network (FCN), U-Net, …
Challenges & Opportunities

• Challenges in previous methods:
  
  • **GPS-only**: noisy, incomplete and inaccurate roadmaps
  
  • **Satellite-only**: easily influenced by dense vegetation, building shadow, dirty roads and so on

• Opportunities: **GPS + Satellite + Deep Learning = ?**
  
  • GPS is increasingly abuentant
  
  • GPS can help satellite images fill the prediction gaps
  
  • Satellite images can help make GPS prediction more accurate
The GPS data we used

- The GPS data
  - 65-taxi and 192-hour data
- The sampling interval is 10s.
  - The spatial resolution is 0.00001 degree of latitude and longitude (about 1m in Beijing).
- We render GPS data in points and lines.
The satellite images we used

• The Satellite Image data:
  • 120 images from *Fifth Ring Road* in Beijing
  • 1024×1024 with resolution of 1m per pixel
  • Labelled manually
  • Road pixel coverage is 13.1%
The model we used

- We used U-Net as skullbone.
- The number of input channels is extended from 3 to 5.
- The input size is 256x256.
Training Details

• Use PyTorch in 2 Nvidia 1080Ti GPUs in about 50 hours

• 5-fold cross validation

• Data enhancement including rotation, flipping, cropping, and HSV tuning

• If the loss does not decrease over 4 epochs, we decrease the learning rate to 1/5 in the next epoch. We terminate training when the loss stops decreasing over 10 epochs or the learning rate is smaller than 1e-7.

• Loss function is  \[ L = (1 - \lambda)L_{ce} - \lambda \log(L_j) \]  , where

\[
L_{ce} = -\frac{1}{N} \sum_{i=0} N_i (y_i \log(y'_i) + (1 - y_i) \log(1 - y'_i)) \\
L_j = \frac{1}{N} \sum_{i=0} N_i \frac{y_i y'_i}{y_i + y'_i - y_i y'_i}
\]
## Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Input</th>
<th>mIoU</th>
<th>Recall</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN</td>
<td>RGB</td>
<td>31.36</td>
<td>38.73</td>
<td>39.24</td>
<td>38.98</td>
</tr>
<tr>
<td></td>
<td>RGB+Line</td>
<td>43.45</td>
<td>60.92</td>
<td>51.51</td>
<td>55.82</td>
</tr>
<tr>
<td></td>
<td>RGB+Point</td>
<td>43.79</td>
<td>64.12</td>
<td>50.59</td>
<td>56.56</td>
</tr>
<tr>
<td></td>
<td>RGB+Line+Point</td>
<td>44.02</td>
<td>64.96</td>
<td>50.48</td>
<td>56.81</td>
</tr>
<tr>
<td>KDE</td>
<td>Line</td>
<td>31.64</td>
<td>39.42</td>
<td>61.61</td>
<td>48.08</td>
</tr>
<tr>
<td></td>
<td>Point</td>
<td>34.06</td>
<td>46.27</td>
<td>56.34</td>
<td>50.82</td>
</tr>
</tbody>
</table>
Results Illustration

true positive = green, false positive = red, false negative = blue
Thanks for your listening

Tongji University