Counting Plants
With Deep Learning

Javier Ribera\textsuperscript{1}, Yuhao Chen\textsuperscript{1},
Christopher Boomsma\textsuperscript{2}, Edward J. Delp\textsuperscript{1}

\textsuperscript{1}Video and Image Processing Laboratory (VIPER)
School of Electrical and Computer Engineering
Purdue University, West Lafayette, Indiana USA

\textsuperscript{2}Department of Agronomy
Purdue University, West Lafayette, Indiana USA
Introduction

• Agronomists and farmers need to know the number of plants in their crops to predict future yield

• Can we count without locating?
• We count plants in a crop field without knowing where they are

• We build our plant dataset from a single image of the entire crop field
• We describe a method to extract images sections or “plots” from an orthorectified image
1. Plot Extraction
Dataset

420 meters ~ 4 football fields
Dataset

1,240 images were extracted

420 meters ~ 4 football fields
Vertical Projection
Profile Function

- Vertical profile over the entire crop field:

\[ \hat{X}_0, \Delta \hat{X} = \arg \min_{x_0, \Delta x} \sum_{n=0}^{N-1} p(x_0 + n\Delta x) \]

- \( N \) : # lines
- \( X_0, \ldots, X_{N-1} \) : Coordinates of each plot-separating line
- \( X_n = X_0 + n\Delta X \quad n = 0, 1, \ldots, N - 1 \)
The Cost Function

- It does not seem appropriate for gradient descent:

\[ \sum_{n=0}^{N-1} p(x_n + n\Delta x) \]

Can be found by brute force

\[ \Delta x \]
1. User provides: (a) number of rows and (b) number of ranges
2. Find range-separating lines:
3. For each range, find row-separating lines:
Method

4. Select the $n$-th row of each range
Method

4. Find range-separating lines for the $n$-th row
Resulting Cropped Images
Resulting Cropped Images
Dataset

• We groundtruthed 2,480 labeled images
  – 80% for training, 10% for validation, 10% for testing

Example image
Plant count: 15
2. Plant Counting
Counting Plants With Deep Learning

Compared CNNs:
- AlexNet-v2
- Inception-v2
- Inception-v3
- Inception-v4

With minimal modification to adapt to image size
Cost Function

• Most research uses cross entropy as cost function, which reduces to

\[ H(p, q) = -\log q(C) \]

where \( q(x) \) are the activations of the last layer, and \( C \) is the true number of plants

• This cost function is not appropriate when the classes are not independent, and there is label noise

• We want to count, not classify

• We propose to use the \( L_p \) norm

\[ L_p (x, \hat{x}) = |x - \hat{x}|^p \]

and test which value of \( p \) provides the lowest error
Network Architectures

- We examined several CNN architectures:
  - Alexnet
  - Inception-v2
  - Inception-v3
  - Inception-v4
- We modify the last layers to be able to process non-rectangular images (of size $546 \times 103$)
Stopping Criteria

Error

Epoch

- Validation
- Test
- Training
Stopping Criteria

![Graph showing error vs. epoch with lines for training, validation, and test sets.]

- **Training**
- **Validation**
- **Test**

**Epoch**
Value For $p$

- Our metric for testing is Mean Average Percentage Error

\[ MAPE = 100 \frac{|\hat{x} - C|}{C} \]

- Effect of $p$ on the error, evaluated using AlexNet

<table>
<thead>
<tr>
<th>$p$</th>
<th>MAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>8.2%</td>
</tr>
<tr>
<td>1.8</td>
<td>8.4%</td>
</tr>
<tr>
<td>1.5</td>
<td>8.5%</td>
</tr>
<tr>
<td>1</td>
<td>7.9%</td>
</tr>
</tbody>
</table>
Results

• Performance of different architectures was evaluated

<table>
<thead>
<tr>
<th>Network</th>
<th>MAPE (w/o data augm)</th>
<th>MAPE (w/ data augm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>8.3%</td>
<td>7.9%</td>
</tr>
<tr>
<td>Inception-v2</td>
<td>8.2%</td>
<td>6.7%</td>
</tr>
<tr>
<td>Inception-v3</td>
<td>7.1%</td>
<td>6.7%</td>
</tr>
<tr>
<td>Inception-v4</td>
<td>12.4%</td>
<td>11.4%</td>
</tr>
</tbody>
</table>

• (using $p=1$)
Conclusions

• We presented a CNN-based method to count plants without locating them

• We presented a method to segment (or extract) image sections, or plots, from an orthorectified image

• Future work will include investigating loss functions more stable than the L1, such as the smooth L1, and training with larger datasets