DetectionEvaluationJ

A tool for measuring the goodness of object detection algorithms

C. Domínguez, M. García, J. Heras, A. Inés, E. Mata, and V. Pascual
Department of Mathematics and Computer Science, University of La Rioja, Spain
Introduction

Object detection algorithms are applied in diverse computer vision applications, surveillance, traffic monitoring, melanoma detection...

In order to evaluate the quality of those algorithms, we compare the detected Regions of interest (ROIs) with the gold standard using different metrics
Introduction
Introduction

Detection algorithm
Introduction

Detection algorithm
Introduction

Gold standard
Introduction

Detection algorithm

Gold standard
Introduction

Detection algorithm

Gold standard

True positive
Introduction

Detection algorithm
Gold standard
False negative
Detection algorithm
Gold standard
False positive
Problems:

- Measuring the performance of object detection algorithms is a common task
- There is not a simple tool to carry out this task automatically
- Measuring the quality of the algorithms manually is not sensible
- Developing ad hoc tools is not a solution either
Goal:

- Develop a simple-to-use tool to evaluate the performance of object detection algorithms using several metrics
ImageJ

- Open-source Java-based image processing program
- Extensible via plugins and recordable macros
- Available at [http://imagej.net/](http://imagej.net/)
DetectionEvaluationJ

ImageJ plugin for measuring the performance of object detection algorithms

Input:
- An image (or set of images)
- The gold standard
- The ROIs obtained by the detection algorithm

Output:
- Report that summarises the quality of the detection algorithm based on several metrics
DetectionEvaluationJ workflow
DetectionEvaluationJ

DetectionEvaluationJ input:

▶ Images
▶ Gold Standard: can be fixed, exported and imported using DetectionEvaluationJ
▶ Detected ROIs:
  ▶ ROIs detected with ImageJ
  ▶ ROIs detected by other programs encoded using ROIXML
DetectionEvaluationJ results

- The user can measure the goodness of the detected regions using the pixel-level metrics
- Several detection algorithms can be loaded to compare their quality
- This plugin can also be applied to study inter-rater agreement among experts
List of measures

- Accuracy
- Precision
- Recall
- Fallout
- Sensitivity
- Negative predictive value
- LR+
- LR-
- Specificity
- False negative rate
- False discovery rate
- F-measures (0.5, 1, 2)
List of measures

- Intersection over union
- Fowlkes Mallows index
- Diagnostic odds ratio
- Balanced accuracy
- Error rate
- Youden’s J statistic
- Markedness
- Matthews correlation coefficient
- ROC space
Example
Example
Example
Example
Example
Example
### Example

#### Measurements

<table>
<thead>
<tr>
<th>Index</th>
<th>Label</th>
<th>True ROI</th>
<th>Hypothesised ROI</th>
<th>True positive</th>
<th>False Positive</th>
<th>True negative</th>
<th>False negative</th>
<th>Positive</th>
<th>Negative</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0279-0115</td>
<td>0279-0115</td>
<td>6785.0</td>
<td>0.0</td>
<td>308651.0</td>
<td>7124.0</td>
<td>13909.0</td>
<td>308651.0</td>
<td>0.9779141865079365</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0279-0115</td>
<td>0279-0130</td>
<td>8479.0</td>
<td>13.0</td>
<td>309638.0</td>
<td>5430.0</td>
<td>13909.0</td>
<td>308651.0</td>
<td>0.9831256200396625</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0279-0115</td>
<td>0251-0118</td>
<td>8885.0</td>
<td>0.0</td>
<td>308651.0</td>
<td>5224.0</td>
<td>13909.0</td>
<td>308651.0</td>
<td>0.9838045634920635</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0279-0115</td>
<td>0258-0119</td>
<td>8478.0</td>
<td>698.0</td>
<td>307953.0</td>
<td>5431.0</td>
<td>13909.0</td>
<td>308651.0</td>
<td>0.9809988392835714</td>
<td></td>
</tr>
</tbody>
</table>

#### Precision

<table>
<thead>
<tr>
<th>Precision</th>
<th>Recall</th>
<th>Fallout</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Negative predictive value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>0.4973136458406225</td>
<td>0.0</td>
<td>0.4973136458406225</td>
<td>1.0</td>
<td>0.9774396326496290</td>
</tr>
<tr>
<td>0.9984691474326780</td>
<td>0.6096052915378531</td>
<td>4.21187664472106E-5</td>
<td>0.6096052915378531</td>
<td>0.9999578312315528</td>
<td>0.9827107565381</td>
</tr>
<tr>
<td>1.0</td>
<td>0.6244158458652017</td>
<td>0.0</td>
<td>0.6244158458652017</td>
<td>1.0</td>
<td>0.9833564317006177</td>
</tr>
<tr>
<td>0.92393199651264170</td>
<td>0.60953339564310880</td>
<td>0.0022614538750886925</td>
<td>0.60953339564310880</td>
<td>0.9977385461249113</td>
<td>0.9825698236029918</td>
</tr>
</tbody>
</table>
Conclusions

Measuring the performance of object detection algorithms is a common problem in computer vision.

DetectionEvaluationJ is an open source ImageJ plugin that solves this problem.
Where can we find this plugin?

DetectionEvaluationJ is freely available at joheras.github.io/DetectionEvaluationJ/
Questions?