DEFCoN-ImageJ Documentation

Release 0.1.3

Baptiste Ottino, Kyle M. Douglass

Jun 05, 2018

Contents

An ImageJ plugin for DEFCoN, the fluorescence spot counter based on fully convolutional neural networks

CHAPTER 1

Instructions

Contents

1.1 Installation

You will need a trained DEFCoN network model to use the plugin regardless of the installation procedure that you choose to follow. These may be downloaded from the DEFCoN-ImageJ Wiki.

1.1.1 Fiji/ImageJ update site

1. Make a backup of your Fiji folder. (This is always a good idea before adding an update site.
2. Open Fiji and navigate to *Help* > *Update...*. Install any updates and restart Fiji if necessary.
3. In the ImageJ Update dialog, click the *Manage update sites* button, scroll to the bottom of the list, and add http://sites.imagej.net/Kmdouglass under the URL column. You may give it any name you want, such as LEB-EPFL.

4. Install all the updates and restart Fiji.
5. Verify that the plugin is recognized by clicking **Plugins** on the menu bar and looking for **DEFCoN**.

1.1.2 Manual installation

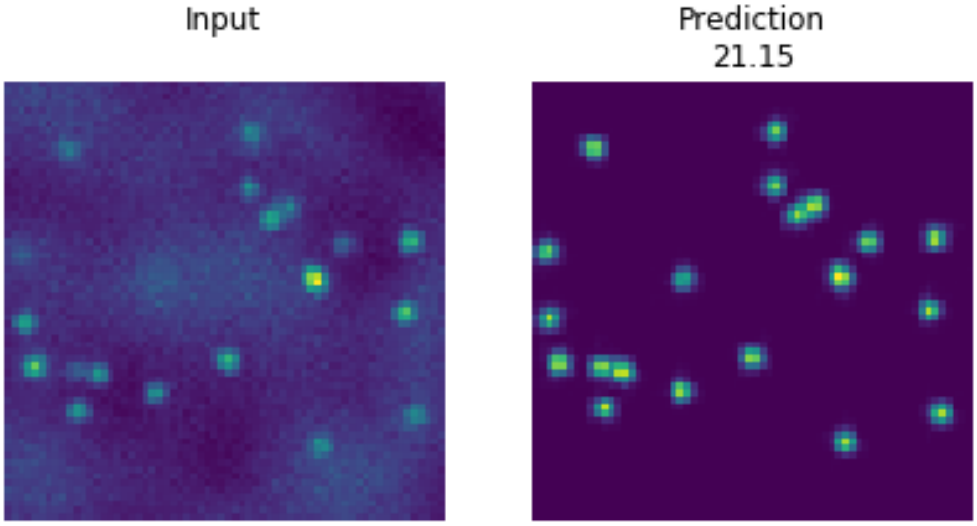
1. Download the latest DEFCoN-ImageJ .jar and dependencies from the GitHub releases.
2. Place the DEFCoN-ImageJ .jar file inside your **<ImageJ_Root>/plugins** folder. Unzip the dependencies into the **<ImageJ_Root>/jars** folder.
3. Verify that the plugin is recognized by opening ImageJ/Fiji by clicking **Plugins** on the menu bar and looking for **DEFCoN**.

1.2 Using DEFCoN

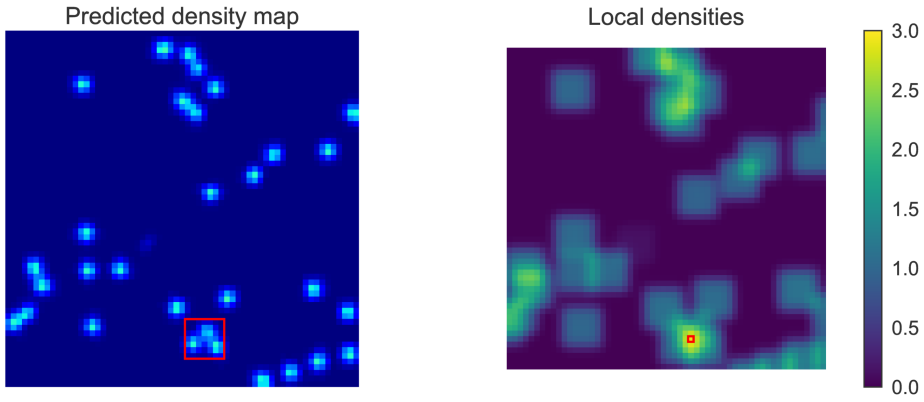
DEFCoN provides two kinds of networks for spot counting. They are called

1. **density map**
2. **maximum local count**

The density map network produces a density map estimate from an image. The sum of the pixel values in a region of the density map is equal to the estimated number of fluorescent spots within that region.



The maximum local count network first computes the sum of the pixels in all possible subregions of the density map. It returns the single largest value of all these sums. In effect, it reports the highest *local* density of spots across an entire image. (The original maximum count network available on the DEFCoN-ImageJ wiki uses 7x7 pixels subregions.)



1.2.1 The ImageJ GUI

1. Open an image or image stack or select a currently open image/stack.
2. Navigate to **Plugins > DEFCoN > Density map...** on the menu bar.
3. A dialog will appear asking for you to enter the name of a folder. This folder should contain a saved TensorFlow model of a DEFCoN density map network. (The dialog will remember the previous path that was used, meaning you will not have to reenter this path every time you wish to use DEFCoN.)
4. Click OK to start processing.

To compute the maximum local count, repeat these steps but select **Plugins > DEFCoN > Maximum local count...** instead. In the dialog, select a folder containing a saved maximum local count network.

1.2.2 ImageJ macros

The macro panel may be accessed through the menu bar by navigating to **Plugins > New > Macro**.

The macros for running the counting network and maximum local count network are:

```
run("Density map...","load=/path/to/density/network");
run("Maximum local count...", "load=/path/to/max/count/network");
```

1.3 Training your own DEFCoN networks

Information on training your own DEFCoN network may be found at the DEFCoN project page. DEFCoN will usually perform best when trained on data that closesly matches your particular use case.

CHAPTER 2

Javadoc

2.1 ch.epfl.leb.defcon.ij

2.1.1 DensityCount

public class **`DensityCount`** implements PlugInFilter

Computes a density map estimate for counting objects within an image.

Author Baptiste Ottino

Methods

run

public void **`run`**(ImageProcessor *ip*)

Computes a density map from the selected image stack. A results table that indicates the count is displayed in addition to the density map image.

Parameters

- **`ip`** – The input image processor.

setup

public int **`setup`**(String *pathToModel*, ImagePlus *imp*)

Sets up the PlugInFilter.

Parameters

- **`pathToModel`** – The path to a saved TensorFlow model bundle.
- **`imp`** – The currently active image.

Returns A flag indicating which types of images this plugin handles.

See also: PlugInFilter

2.1.2 MaxCountFCN

public class **`MaxCountFCN`** extends *AbstractPredictor* implements PlugInFilter

Methods

run

public void **`run`** (ImageProcessor *ip*)

setup

public int **`setup`** (String *pathToModel*, ImagePlus *imp*)
Sets up the PlugInFilter. Please see for more information on the PlugInFilter API.

Parameters

- **`pathToModel`** – The path to a saved TensorFlow model bundle.
- **`imp`** – The currently active image.

Returns A flag indicating which types of images this plugin handles.

See also: PlugInFilter

2.2 ch.epfl.leb.defcon.ij.gui

2.2.1 RunDensityCount

public class **`RunDensityCount`** implements PlugIn
Launches the density count DEFCoN plugin.

Author Kyle M. Douglass

Methods

run

public void **`run`** (String *arg*)

2.2.2 RunMaxCountFCN

public class **`RunMaxCountFCN`** implements PlugIn
Launches the maximum local count DEFCoN plugin.

Author Kyle M. Douglass

Methods

run

public void **run** (String *arg*)

2.3 ch.epfl.leb.defcon.predictors

2.3.1 ImageBitDepthException

public class **ImageBitDepthException** extends Exception
Raised when an image with an invalid bit-depth is supplied to the predictor.

Author Kyle M. Douglass

Constructors

ImageBitDepthException

public **ImageBitDepthException** (String *msg*)

2.3.2 NoLocalCountMapException

public class **NoLocalCountMapException** extends Exception
Raised when a predictor has not yet made a local count map estimation.

Author Kyle M. Douglass

Constructors

NoLocalCountMapException

public **NoLocalCountMapException** (String *msg*)

2.3.3 Predictor

public interface **Predictor**
Makes density map predictions from images.

Author Kyle M. Douglass

Methods

close

public void **close** ()
Closes resources associated with this predictor.

getCount

public double **getCount**()
Returns the most recent count.

Throws

- *ch.epfl.leb.defcon.predictors.UninitializedPredictorException* –

Returns The predicted count from the density map.

getDensityMap

public FloatProcessor **getDensityMap**()
Returns the most recently calculated density map prediction.

Throws

- *ch.epfl.leb.defcon.predictors.UninitializedPredictorException* –

Returns The predicted density map.

getLocalCountMap

public FloatProcessor **getLocalCountMap**()
Returns the most recently-calculated local count map.

Throws

- *ch.epfl.leb.defcon.predictors.NoLocalCountMapException* –

Returns The most recently calculated local count map.

getMaximumLocalCount

public double **getMaximumLocalCount**(int *boxSize*)
Returns the maximum local count value. This value is obtained by convolving the density map with a square kernel whose values are all 1 and then taking the maximum of the resulting map. It effectively produces the highest count value over length scales equal to the size of the kernel.

Parameters

- **boxSize** – The width of the square kernel.

Throws

- *ch.epfl.leb.defcon.predictors.UninitializedPredictorException* –

Returns The maximum local count from the density map.

predict

public void **predict** (ImageProcessor *ip*)
Makes a density map prediction from a 2D image.

Parameters

- **ip** – The image to perform a prediction on.

setup

public void **setup** (String *pathToModel*)
Initializes the predictor with a saved TensorFlow model bundle.

Parameters

- **pathToModel** – The path to a saved TensorFlow model bundle.

2.3.4 SessionClosedException

public class **SessionClosedException** extends Exception
Raised when a prediction is requested but the TensorFlow session has already been closed.

Author Kyle M. Douglass

Constructors

SessionClosedException

public **SessionClosedException** (String *msg*)

2.3.5 UninitializedPredictorException

public class **UninitializedPredictorException** extends Exception
Raised when a predictor has not yet been fully initialized.

Author Kyle M. Douglass

Constructors

UninitializedPredictorException

public **UninitializedPredictorException** (String *msg*)

2.4 ch.epfl.leb.defcon.predictors.internal

2.4.1 AbstractPredictor

public abstract class **AbstractPredictor**
A base implementation for DEFCoN predictors. A predictor is a class that takes an image(s) as input and

estimates a density map of object locations from it.

Author Kyle M. Douglass

Fields

isClosed

protected boolean **`isClosed`**
Has the TensorFlow session been closed?

tfSession

protected Session **`tfSession`**
A copy of the current TensorFlow session.

Methods

close

public void **`close`**()
Closes resources associated with this predictor.

finalize

protected void **`finalize`**()
Failsafe in case the TensorFlow session has not been closed.

Throws

- **`java.lang.Throwable`** –

imageToTensor

protected static Tensor<Float> **`imageToTensor`**(ImagePlus *imp*)
Converts an ImageJ image to a TensorFlow tensor. The original image is preserved.

Parameters

- **`imp`** – The image to convert.

Returns A tensor representing the data in the original image.

setup

public void **`setup`**(String *pathToModel*)
Initializes the predictor.

Parameters

- **`pathToModel`** – The path to a saved TensorFlow model bundle.

2.4.2 DefaultPredictor

public class **DefaultPredictor** extends *AbstractPredictor* implements *Predictor*
Makes density map predictions from images.

Author Baptiste Ottino, Kyle M. Douglass

Methods

finalize

protected void **finalize**()
Failsafe in case the TensorFlow session has not been closed.

Throws

- **java.lang.Throwable** –

getCount

public double **getCount**()
Returns the most recent count.

Throws

- ***ch.epfl.leb.defcon.predictors.UninitializedPredictorException*** –

Returns The predicted count from the density map.

getDensityMap

public FloatProcessor **getDensityMap**()
Returns the most recently calculated density map prediction.

Throws

- ***ch.epfl.leb.defcon.predictors.UninitializedPredictorException*** –

Returns The predicted density map.

getLocalCountMap

public FloatProcessor **getLocalCountMap**()
Returns the most recently-calculated local count map.

Throws

- ***ch.epfl.leb.defcon.predictors.NoLocalCountMapException*** –

Returns The most recently calculated local count map.

getMaximumLocalCount

public double **getMaximumLocalCount** (int *boxSize*)

Returns the maximum local count value. This value is obtained by convolving the density map with a square kernel whose values are all 1 and then taking the maximum of the resulting map. It effectively produces the highest count value over length scales equal to the size of the kernel.

Parameters

- **boxSize** – The width of the square kernel.

Throws

- *ch.epfl.leb.defcon.predictors.UninitializedPredictorException* –

Returns The maximum local count from the density map.

predict

public void **predict** (ImageProcessor *ip*)

Makes a density map prediction from a 2D image. If either of the image's width or height is not divisible by 4, they will be cropped to the next largest multiple of four.

Parameters

- **ip** – The image to perform a prediction on.

Throws

- *ch.epfl.leb.defcon.predictors.ImageBitDepthException* –
- *ch.epfl.leb.defcon.predictors.SessionClosedException* –

2.4.3 DefaultPredictorIT

public class **DefaultPredictorIT**

Integration tests for the DefaultPredictor class.

Author Kyle M. Douglass

Constructors

DefaultPredictorIT

public **DefaultPredictorIT** ()

Methods

setUp

public void **setUp** ()

Sets up the integration test.

testGetCount

public void **testGetCount**()
Test of getCount method, of class DefaultPredictor.

testGetDensityMap

public void **testGetDensityMap**()
Test of getDensityMap method, of class DefaultPredictor.

testGetDensityMapResized

public void **testGetDensityMapResized**()
Test of getDensityMap method, of class DefaultPredictor. This test verifies that images whose dimensions are not multiples of four are automatically cropped before computing the density map.

testGetLocalCountMap

public void **testGetLocalCountMap**()
Test of getLocalCountMap, of class DefaultPredictor.

Throws

- **java.lang.Exception** –

testGetMaximumLocalCount

public void **testGetMaximumLocalCount**()
Test of getMaximumLocalCount, of class DefaultPredictor.

Throws

- **java.lang.Exception** –

testPredict

public void **testPredict**()
Test of predict method, of class DefaultPredictor.

2.5 ch.epfl.leb.defcon.utils

2.5.1 GraphBuilder

public class **GraphBuilder**
Appends various operations to TensorFlow graphs.

Author Baptiste Ottino

See also: Output | TensorFlow, Graph | TensorFlow

Methods

constant

public static <T> Output<T> **constant**(Graph *g*, String *name*, Object *value*, Class<T> *type*)
Builds an Output tensor with a single scalar value.

Parameters

- **<T>** –
- **g** – The TensorFlow graph to modify.
- **name** – The full name of the operation.
- **value** – The value output by the operation.
- **type** – The output datatype.

Returns Symbolic handle to the tensor produced by the appended operation.

constant

public static Output<Integer> **constant**(Graph *g*, String *name*, int *value*)
Builds an Output tensor with a single scalar value.

Parameters

- **g** – The TensorFlow graph to modify.
- **name** – The full name of the operation.
- **value** – The value output by the operation.

Returns Symbolic handle to the tensor produced by the appended operation.

expandDims

public static <T> Output<T> **expandDims**(Graph *g*, String *name*, Output<T> *input*, Output<Integer> *dim*)
Adds a dimension to a TensorFlow Output tensor. The dimension as added at the position "dim." (with numpy/tensorflow syntax: 0 for the first position, -1 in the end)

Parameters

- **<T>** –
- **g** – The TensorFlow graph to modify.
- **name** – The full name of the operation.
- **input** – The input operation to expand.
- **dim** –

Returns Symbolic handle to the tensor produced by the appended operation.

This is the ImageJ plugin for the density estimation by fully convolutional networks (DEFCoN) algorithm, an image processing tool for fluorescence spot counting. With this plugin, you can use trained DEFCoN models directly on images from within ImageJ.

CHAPTER 3

Authors

- Baptiste Ottino
- Kyle M. Douglass

CHAPTER 4

See also

- DEFCoN-ImageJ Wiki - Download site for pre-trained DEFCoN models
- DEFCoN - Software for training a DEFCoN model
- ALICA - Automated laser illumination control for Micro-Manager
- SASS - SMLM acquistion simulation software

CHAPTER 5

Indices and tables

- genindex
- search

Index