My Project

Generated by Doxygen 1.8.9.1

Thu Mar 10 2016 20:23:41

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Namespace Documentation

4.1 Box Namespace Reference

Classes

- class Box

4.2 end2end Namespace Reference

Classes

- class EndToEnd

Functions

- def main ()

4.2.1 Function Documentation

4.2.1.1 def end2end.main ( )

4.3 EvalOneToMany Namespace Reference

Classes

- class EvalOneToMany

Functions

- def main ()

Variables

- int NUM_CLASS = 3
- float ONETOONE_THRESHOLD = 0.85
- float ONETOMANY_THRESHOLD = 0.1
- list LABELS = ['text', 'title', 'other']

4.3.1 Function Documentation

4.3.1.1 def EvalOneToMany.main ( )

4.3.2 Variable Documentation

4.3.2.1 list EvalOneToMany.LABELS = ['text', 'title', 'other']

4.3.2.2 int EvalOneToMany.NUM_CLASS = 3

4.3.2.3 float EvalOneToMany.ONETOMANY_THRESHOLD = 0.1

4.3.2.4 float EvalOneToMany.ONETOONE_THRESHOLD = 0.85

4.4 Evaltest Namespace Reference

Classes

- class EvalOneToMany

Functions

- def main ()

Variables

- int NUM_CLASS = 3
- float ONETOONE_THRESHOLD = 0.85
- float ONETOMANY_THRESHOLD = 0.1
- list LABELS = ['text', 'title', 'other']

4.4.1 Function Documentation

4.4.1.1 def Evaltest.main ( )

4.4.2 Variable Documentation

4.4.2.1 list Evaltest.LABELS = ['text', 'title', 'other']

4.4.2.2 int Evaltest.NUM_CLASS = 3

4.4.2.3 float Evaltest.ONETOMANY_THRESHOLD = 0.1

4.4.2.4 float Evaltest.ONETOONE_THRESHOLD = 0.85

4.5 IdentityMetric Namespace Reference

Classes

- class EvalOneToMany

Functions

- def main ()

Variables

- int NUM_CLASS = 3
- float ONETOONE_THRESHOLD = 0.85
- float ONETOMANY_THRESHOLD = 0.1
- list LABELS = ['text', 'title', 'other']

4.5.1 Function Documentation

4.5.1.1 def IdentityMetric.main ( )

4.5.2 Variable Documentation

4.5.2.1 list IdentityMetric.LABELS = ['text', 'title', 'other']

4.5.2.2 int IdentityMetric.NUM_CLASS = 3

4.5.2.3 float IdentityMetric.ONETOMANY_THRESHOLD = 0.1

4.5.2.4 float IdentityMetric.ONETOONE_THRESHOLD = 0.85

4.6 Image Namespace Reference

Classes

- class NewsImage

4.7 imageBS Namespace Reference

Variables

- list f_image = sys.argv[1]
- list f_xml = sys.argv[2]
- list f_out = sys.argv[3]
- tuple f = f_image.split('/')
- string f_csv = 'images/'
- list boxes = [ ]
- tuple dat = line.strip('\n')
- dictionary seg = {"annotations":[ ]}
- int id = 0
- dictionary tbInfo = {}

4.7.1 Detailed Description

```
python 2.7
```

4.7.2 Variable Documentation

4.7.2.1 list imageBS.boxes = [ ]

4.7.2.2 tuple imageBS.dat = line.strip('\n')

4.7.2.3 tuple imageBS.f = f_image.split('/')

4.7.2.4 string imageBS.f_csv = 'images/'

4.7.2.5 list imageBS.f_image = sys.argv[1]

4.7.2.6 list imageBS.f_out = sys.argv[3]

4.7.2.7 list imageBS.f_xml = sys.argv[2]

4.7.2.8 int imageBS.id = 0

4.7.2.9 dictionary imageBS.seg = {"annotations":[ ]}

4.7.2.10 dictionary imageBS.tblnfo = {}

4.8 Latex Namespace Reference

Functions

- def replace_dict (string, replacements)
- def gen_latex (title, fnameout)
- def gen_pdf (fnametex, fnamepdf)

Variables

- tuple latex_template = f.read()

4.8.1 Function Documentation

4.8.1.1 def Latex.gen_latex (*title, fnameout*)

4.8.1.2 def Latex.gen_pdf (*fnametex, fnamepdf*)

4.8.1.3 def Latex.replace_dict (*string, replacements*)

4.8.2 Variable Documentation

4.8.2.1 tuple Latex.latex_template = f.read()

4.9 pipe_to_jason Namespace Reference

Variables

- list polys = [l.split('|') for l in f.read().split('\n') if l]
- list labpolys = [(p[0],eval('[%s]'%(','.join(p[1:])))) for p in polys]
- int sh = 6351

- int sw = 3960
- dictionary json

4.9.1 Variable Documentation

4.9.1.1 dictionary pipe_to_jason.json

Initial value:

```
1 = {'annotations': [
2     {'id':pi,'height':(b[1][0]-b[0][0])*sh,'width':(b[1][1]-b[0][1])*sw,'y':(b[0][0])*sh,'x':b[0][1]*sw, '
      type':'rect', 'class':lab}
3     for pi, (lab,pol) in enumerate(labpolys) for b in pol]}
```

4.9.1.2 list pipe_to_jason.labpolys = [(p[0],eval('[%s]'%(','.join(p[1:])))) for p in polys]

4.9.1.3 list pipe_to_jason.polys = [l.split('|') for l in f.read().split('\n') if l]

4.9.1.4 int pipe_to_jason.sh = 6351

4.9.1.5 int pipe_to_jason.sw = 3960

4.10 Polygon Namespace Reference

Classes

- class Polygon

4.11 processXML Namespace Reference

Classes

- class ProcessXML

4.11.1 Detailed Description

python 2.7

4.12 readJSON Namespace Reference

Functions

- def seg_from_json (fname)

4.12.1 Detailed Description

python 2.7

4.12.2 Function Documentation

4.12.2.1 def readJSON.seg_from_json (*fname*)

4.13 Segmentation Namespace Reference

Classes

- class Segmentation

4.14 testseg Namespace Reference

Functions

- def size_of_image (imname)

Variables

- list f_image = sys.argv[1]
- list f_xml = sys.argv[2]
- list f_out = sys.argv[3]
- tuple pxml = processXML.ProcessXML(f_xml)
- tuple tbList = pxml.getTBData()
- dictionary seg = {"annotations":[ ]}
- int id = 0
- dictionary tbInfo = {}

4.14.1 Detailed Description

```
python 2.7
```

4.14.2 Function Documentation

4.14.2.1 def testseg.size_of_image (*imname*)

4.14.3 Variable Documentation

4.14.3.1 list testseg.f_image = sys.argv[1]

4.14.3.2 list testseg.f_out = sys.argv[3]

4.14.3.3 list testseg.f_xml = sys.argv[2]

4.14.3.4 int testseg.id = 0

4.14.3.5 tuple testseg.pxml = processXML.ProcessXML(f_xml)

4.14.3.6 dictionary testseg.seg = {"annotations":[ ]}

4.14.3.7 dictionary testseg.tbInfo = {}

4.14.3.8 list testseg.tbList = pxml.getTBData()

4.15 textblocksBS Namespace Reference

Functions

- def size_of_image (imname)

Variables

- list f_image = sys.argv[1]
- list f_xml = sys.argv[2]
- list f_out = sys.argv[3]
- tuple pxml = processXML.ProcessXML(f_xml)
- tuple tbList = pxml.getTBData()
- dictionary seg = {"annotations":[ ]}
- int id = 0
- dictionary tbInfo = {}

4.15.1 Detailed Description

```
python 2.7
```

4.15.2 Function Documentation

4.15.2.1 def textblocksBS.size_of_image (*imname*)

4.15.3 Variable Documentation

4.15.3.1 list textblocksBS.f_image = sys.argv[1]

4.15.3.2 list textblocksBS.f_out = sys.argv[3]

4.15.3.3 list textblocksBS.f_xml = sys.argv[2]

4.15.3.4 int textblocksBS.id = 0

4.15.3.5 tuple textblocksBS.pxml = processXML.ProcessXML(f_xml)

4.15.3.6 dictionary textblocksBS.seg = {"annotations":[ ]}

4.15.3.7 dictionary textblocksBS.tbInfo = {}

4.15.3.8 list textblocksBS.tbList = pxml.getTBData()

Chapter 5

Class Documentation

5.1 Box.Box Class Reference

Public Member Functions

- def __init__
- def from_point (self, coor)
- def from_corners (self, coorA, coorB)
- def flatten (self)
- def __str__ (self)
- def from_str (self, string)
- def area
- def center (self)
- def disconnect_distance (self, other)
- def overlaps (self, other)
- def join (self, other)
- def intersect (self, other)

Public Attributes

- points

5.1.1 Detailed Description

```
A box is an axis-aligned rectangle, represented by [mincoors, maxcoors]
    For us, coordinates are length-2 lists of form [y, x], in units of
    pixels of the original image.
```

5.1.2 Constructor & Destructor Documentation

5.1.2.1 def Box.Box.__init__ (*self*, *coor0* = None, *coor1* = None, *string* = None)

```
The 0th coor is [miny,minx]; 1st coor is [maxy,maxx];
   we may also initialize from a string, e.g. '[[1.0,0.0],[2.0,3.0]]'.
   If only coor0 is specified, then a point [coor0,coor0] is created.

   Member variables:
   0. coors represents coordinates, e.g. [[1.0,0.0],[2.0,3.0]].
```

5.1.3 Member Function Documentation

5.1.3.1 def Box.Box.__str__ (*self*)

```
string representation of box is simply that of its coordinates,
   e.g. '[[1.0,0.0],[2.0,3.0]]'
```

5.1.3.2 def Box.Box.area (*self, image* = None)

```
returns image area, weighted by image-values if an image is given
```

5.1.3.3 def Box.Box.center (*self*)

```
returns [centery,centerx]
```

5.1.3.4 def Box.Box.disconnect_distance (*self, other*)

```
returns the largest possible rectangle --- infinite in one of
   the vertical or horizontal directions --- that separates the
   two boxes. if the boxes overlap, this margin will be negative.
```

5.1.3.5 def Box.Box.flatten (*self*)

```
converts internal [[miny,minx],[maxy,maxx]] into [miny,minx,maxy,maxx]
```

5.1.3.6 def Box.Box.from_corners (*self, coorA, coorB*)

```
Constructs the box from any two nonadjacent corners, in any order.
   Canonicalizes the coordinates into [mincoors,maxcoors] form.
```

5.1.3.7 def Box.Box.from_point (*self, coor*)

```
initializes box as breadthless, lengthless, and
   centered at the inputted coordinate
```

5.1.3.8 def Box.Box.from_str (*self, string*)

5.1.3.9 def Box.Box.intersect (*self, other*)

```
returns the largest box contained in both inputs,
   or else the [[0,0],[0,0]] box if there's no shared area
```

5.1.3.10 def Box.Box.join (*self, other*)

```
returns the smallest box containing both inputs
```

5.1.3.11 def Box.Box.overlaps (*self, other*)

```
Returns whether do the boxes share geometric area strictly greater than 0?
   We don't consider image-values here.
```

5.1.4 Member Data Documentation

5.1.4.1 Box.Box.points

The documentation for this class was generated from the following file:

- Box.py

5.2 end2end.EndToEnd Class Reference

Public Member Functions

- def __init__
- def segment (self)
- def evaluate (self)
- def collect_data (self)
- def plot_performance_curve (self)
- def generate_report (self)

Public Attributes

- metrics
- implementations
- files
- eval_results

5.2.1 Detailed Description

```
assumes segmentation algo.s take command line arguments as follows:
     python dilate_segmenter.py <imagename> <xmlname> <outname>
  e.g.
     python dilate_segmenter.py 0005.jp2 0005.xml 0005_seg.json
```

5.2.2 Constructor & Destructor Documentation

5.2.2.1 def end2end.EndToEnd.__init__ (*self, config_filename =* `'config.txt'`)

5.2.3 Member Function Documentation

5.2.3.1 def end2end.EndToEnd.collect_data (*self*)

```
read in output data from evaluation metrics
    e.g. assume we have an image file named 0005.jpg, and segmentation algorithm
    textblocksBS.py, then this function will read data from 0005.jpg.textblcoksBS.py
    data will be saved in a dictionary in the following format:
    {"textblocksBS.py":
{
    precision:
    recall:
    score:
```

```
    history:
    {
        (precision1, recall1, score1),
        (precision2, recall2, score2)
    }
}
```

5.2.3.2 def end2end.EndToEnd.evaluate (*self*)

```
assumes format groundtruth - ....gt.json
```

5.2.3.3 def end2end.EndToEnd.generate_report (*self*)

5.2.3.4 def end2end.EndToEnd.plot_performance_curve (*self*)

5.2.3.5 def end2end.EndToEnd.segment (*self*)

5.2.4 Member Data Documentation

5.2.4.1 end2end.EndToEnd.eval_results

5.2.4.2 end2end.EndToEnd.files

5.2.4.3 end2end.EndToEnd.implementations

5.2.4.4 end2end.EndToEnd.metrics

The documentation for this class was generated from the following file:

- end2end.py

5.3 Evaltest.EvalOneToMany Class Reference

Public Member Functions

- def __init__
- def evaluate (self)
- def save_output (self, record)

Public Attributes

- seg_path
- gt_path
- xml_path
- img_path
- history_path
- ground_truth
- seg_to_eval

Static Public Attributes

- string gt_path = './images/0005.jpg.gt.json'
- string seg_path = './images/0005.jpg.json'
- string img_path = './images/0005.jpg'
- string xml_path = './images/0005.jpg.xml'
- string history_path = './images/0005.jpg.evalout'
- string imp_name = 'textblocksBS.py'
- list eval_history = [ ]

5.3.1 Constructor & Destructor Documentation

5.3.1.1 def Evaltest.EvalOneToMany.__init__ (*self, gt_path, seg_path, img_path* = None, *xml_path* = None, *imp_name* = None)

5.3.2 Member Function Documentation

5.3.2.1 def Evaltest.EvalOneToMany.evaluate (*self*)

5.3.2.2 def Evaltest.EvalOneToMany.save_output (*self, record*)

5.3.3 Member Data Documentation

5.3.3.1 list Evaltest.EvalOneToMany.eval_history = [ ] [static]

5.3.3.2 Evaltest.EvalOneToMany.ground_truth

5.3.3.3 string Evaltest.EvalOneToMany.gt_path = './images/0005.jpg.gt.json' [static]

5.3.3.4 Evaltest.EvalOneToMany.gt_path

5.3.3.5 string Evaltest.EvalOneToMany.history_path = './images/0005.jpg.evalout' [static]

5.3.3.6 Evaltest.EvalOneToMany.history_path

5.3.3.7 string Evaltest.EvalOneToMany.img_path = './images/0005.jpg' [static]

5.3.3.8 Evaltest.EvalOneToMany.img_path

5.3.3.9 string Evaltest.EvalOneToMany.imp_name = 'textblocksBS.py' [static]

5.3.3.10 string Evaltest.EvalOneToMany.seg_path = './images/0005.jpg.json' [static]

5.3.3.11 Evaltest.EvalOneToMany.seg_path

5.3.3.12 Evaltest.EvalOneToMany.seg_to_eval

5.3.3.13 string Evaltest.EvalOneToMany.xml_path = './images/0005.jpg.xml' [static]

5.3.3.14 Evaltest.EvalOneToMany.xml_path

The documentation for this class was generated from the following file:

- Evaltest.py

5.4 IdentityMetric.EvalOneToMany Class Reference

Public Member Functions

- def __init__
- def evaluate (self)
- def MatchScore
- def clear_history (self)
- def plot_performance_curve (self)
- def plot_result (self)
- def generate_report (self)

Public Attributes

- image_folder
- ground_truth
- seg_to_eval
- eval_history

Static Public Attributes

- string image_path = './images/0005.jpg'
- string segmentation_path = './images/0005.jpg.demo_full.txt'
- string ground_truth_path = './images/0005.jpg.demo_full.txt'
- image = None
- list eval_history = [ ]

5.4.1 Constructor & Destructor Documentation

5.4.1.1 def IdentityMetric.EvalOneToMany.__init__ (*self, gt_path, seg_path, img_path* = `None`, *xml_path* = `None`)

5.4.2 Member Function Documentation

5.4.2.1 def IdentityMetric.EvalOneToMany.clear_history (*self*)

5.4.2.2 def IdentityMetric.EvalOneToMany.evaluate (*self*)

5.4.2.3 def IdentityMetric.EvalOneToMany.generate_report (*self*)

5.4.2.4 def IdentityMetric.EvalOneToMany.MatchScore (*self, Box1, Box2, use_black_pixel* = `False`)

5.4.2.5 def IdentityMetric.EvalOneToMany.plot_performance_curve (*self*)

5.4.2.6 def IdentityMetric.EvalOneToMany.plot_result (*self*)

5.4.3 Member Data Documentation

5.4.3.1 list IdentityMetric.EvalOneToMany.eval_history = [ ] `[static]`

5.4.3.2 IdentityMetric.EvalOneToMany.eval_history

5.4.3.3 IdentityMetric.EvalOneToMany.ground_truth

5.4.3.4 string IdentityMetric.EvalOneToMany.ground_truth_path = './images/0005.jpg.demo_full.txt' [static]

5.4.3.5 IdentityMetric.EvalOneToMany.image = None [static]

5.4.3.6 IdentityMetric.EvalOneToMany.image_folder

5.4.3.7 string IdentityMetric.EvalOneToMany.image_path = './images/0005.jpg' [static]

5.4.3.8 IdentityMetric.EvalOneToMany.seg_to_eval

5.4.3.9 string IdentityMetric.EvalOneToMany.segmentation_path = './images/0005.jpg.demo_full.txt' [static]

The documentation for this class was generated from the following file:

- IdentityMetric.py

5.5 EvalOneToMany.EvalOneToMany Class Reference

Public Member Functions

- def __init__
- def evaluate (self)
- def save_output (self, record)

Public Attributes

- seg_path
- gt_path
- xml_path
- img_path
- history_path
- ground_truth
- seg_to_eval

Static Public Attributes

- string gt_path = '../../images/0005.jpg.gt.json'
- string seg_path = '../../images/0005.jpg.json'
- string img_path = '../../images/0005.jpg'
- string xml_path = '../../images/0005.jpg.xml'
- string history_path = '../../images/0005.jpg.evalout'
- string imp_name = 'textblocksBS.py'
- list eval_history = [ ]

5.5.1 Constructor & Destructor Documentation

5.5.1.1 def EvalOneToMany.EvalOneToMany.__init__ (*self*, *gt_path*, *seg_path*, *img_path* = None, *xml_path* = None, *imp_name* = None)

5.5.2 Member Function Documentation

5.5.2.1 def EvalOneToMany.EvalOneToMany.evaluate (*self*)

5.5.2.2 def EvalOneToMany.EvalOneToMany.save_output (*self, record*)

5.5.3 Member Data Documentation

5.5.3.1 list EvalOneToMany.EvalOneToMany.eval_history = [ ] [static]

5.5.3.2 EvalOneToMany.EvalOneToMany.ground_truth

5.5.3.3 string EvalOneToMany.EvalOneToMany.gt_path = '../../images/0005.jpg.gt.json' [static]

5.5.3.4 EvalOneToMany.EvalOneToMany.gt_path

5.5.3.5 string EvalOneToMany.EvalOneToMany.history_path = '../../images/0005.jpg.evalout' [static]

5.5.3.6 EvalOneToMany.EvalOneToMany.history_path

5.5.3.7 string EvalOneToMany.EvalOneToMany.img_path = '../../images/0005.jpg' [static]

5.5.3.8 EvalOneToMany.EvalOneToMany.img_path

5.5.3.9 string EvalOneToMany.EvalOneToMany.imp_name = 'textblocksBS.py' [static]

5.5.3.10 string EvalOneToMany.EvalOneToMany.seg_path = '../../images/0005.jpg.json' [static]

5.5.3.11 EvalOneToMany.EvalOneToMany.seg_path

5.5.3.12 EvalOneToMany.EvalOneToMany.seg_to_eval

5.5.3.13 string EvalOneToMany.EvalOneToMany.xml_path = '../../images/0005.jpg.xml' [static]

5.5.3.14 EvalOneToMany.EvalOneToMany.xml_path

The documentation for this class was generated from the following file:

- EvalOneToMany.py

5.6 Image.NewsImage Class Reference

Public Member Functions

- def __init__
- def read_blackness (self)
- def get_total_blackness (self, y0, x0, y1, x1)

Public Attributes

- filenames
- gamma
- blacknesses

Static Public Attributes

- float gamma_default = lambdax:1.0

5.6.1 Detailed Description

```
Processes image .jpgs and .xmls for evaluation pipeline.
   For example, pre-computes area blacknesses for more efficient
   evaluation.
```

5.6.2 Constructor & Destructor Documentation

5.6.2.1 def Image.NewsImage.__init__ (*self, root, gamma* = gamma_default)

```
The argument 'root' could be, for example, root='../Data/0005',
   on which we append '.jpg' and '.xml'.

   member functions:
   0. 'gamma' transforms a grayscale value into a blackness measure.
```

5.6.3 Member Function Documentation

5.6.3.1 def Image.NewsImage.get_total_blackness (*self, y0, x0, y1, x1*)

5.6.3.2 def Image.NewsImage.read_blackness (*self*)

```
presumes .jpg to be grayscale; precomputes area blacknesses
   for more efficient evaluation
```

5.6.4 Member Data Documentation

5.6.4.1 Image.NewsImage.blacknesses

5.6.4.2 Image.NewsImage.filenames

5.6.4.3 Image.NewsImage.gamma

5.6.4.4 float Image.NewsImage.gamma_default = lambdax:1.0 [static]

The documentation for this class was generated from the following file:

- Image.py

5.7 Polygon.Polygon Class Reference

Public Member Functions

- def __init__
- def check_valid (self)
- def area (self)
- def intersect (self, other)
- def __str__ (self)
- def from_str (self, string)
- def create_jaccard

Public Attributes

- label
- boxes

Static Public Attributes

- weight_image = None
- tuple jaccard_precision
- tuple jaccard_recall
- tuple jaccard_similarity

5.7.1 Detailed Description

```
A Polygon is a list of Boxes, presumed pairwise non-overlapping.
    The whole Polygon may be labeled with a `label` in ('text', 'title', etc.).
    Polygon also supports image-weighted areas: if `weight_image`, a member
    of the class on whole, is set to a NewsImage object, then Polygons'
    area-calculations will be weighted by that object's blackness values.
```

5.7.2 Constructor & Destructor Documentation

5.7.2.1 def Polygon.Polygon.__init__ (*self, boxes* = [], *label* = 'text', *string* = None)

5.7.3 Member Function Documentation

5.7.3.1 def Polygon.Polygon.__str__ (*self*)

5.7.3.2 def Polygon.Polygon.area (*self*)

5.7.3.3 def Polygon.Polygon.check_valid (*self*)

```
ensures no overlaps
```

5.7.3.4 def Polygon.Polygon.create_jaccard (*denom_func, docstring* = None)

```
returns a method to compute overlaps of form intersect/denominator
```

5.7.3.5 def Polygon.Polygon.from_str (*self, string*)

5.7.3.6 def Polygon.Polygon.intersect (*self, other*)

5.7.4 Member Data Documentation

5.7.4.1 Polygon.Polygon.boxes

5.7.4.2 tuple Polygon.Polygon.jaccard_precision [static]

Initial value:

```
1 = create_jaccard(lambda s,t,i: s.area(),
2        docstring=)
```

5.7.4.3 tuple Polygon.Polygon.jaccard_recall [static]

Initial value:

```
1 = create_jaccard(lambda s,t,i: t.area(),
2        docstring=)
```

5.7.4.4 tuple Polygon.Polygon.jaccard_similarity [static]

Initial value:

```
1 = create_jaccard(lambda s,t,i: s.area()+t.area()-i,
2         docstring=)
```

5.7.4.5 Polygon.Polygon.label

5.7.4.6 Polygon.Polygon.weight_image = None [static]

The documentation for this class was generated from the following file:

- Polygon.py

5.8 processXML.ProcessXML Class Reference

Public Member Functions

- def __init__ (self, fname)
- def getTIFdimensions (self)
- def getTextblocks (self)
- def getTextLines (self, textblock)
- def getStrings (self, textline)
- def getSpaces (self, textline)
- def writeStSpData (self, wname)
- def writeTBData (self, wname)
- def getTBData (self)

Public Attributes

- filename
- textblocks

5.8.1 Detailed Description

```
class for processing .xml files to get ocr information
```

5.8.2 Constructor & Destructor Documentation

5.8.2.1 def processXML.ProcessXML.__init__ (*self, fname*)

5.8.3 Member Function Documentation

5.8.3.1 def processXML.ProcessXML.getSpaces (*self, textline*)

```
return a list of spaces from a textline
```

5.8.3.2 def processXML.ProcessXML.getStrings (*self, textline*)

```
return list of strings from a textline
```

5.8.3.3 def processXML.ProcessXML.getTBData (*self*)

```
returns a list of (hpos,vpos,width,height) info for each textblock
```

5.8.3.4 def processXML.ProcessXML.getTextblocks (*self*)

```
create list of textblocks
```

5.8.3.5 def processXML.ProcessXML.getTextLines (*self, textblock*)

```
loop thru textlines in a textblock
```

5.8.3.6 def processXML.ProcessXML.getTIFdimensions (*self*)

```
returns (height, width) of the .tif data in the xml file. we will use this
data to change scales to the .jp2 image size
```

5.8.3.7 def processXML.ProcessXML.writeStSpData (*self, wname*)

```
write the coordinate data of strings and spaces to wname file
```

5.8.3.8 def processXML.ProcessXML.writeTBData (*self, wname*)

```
write the coordinate data of textblocks to wname file
```

5.8.4 Member Data Documentation

5.8.4.1 processXML.ProcessXML.filename

5.8.4.2 processXML.ProcessXML.textblocks

The documentation for this class was generated from the following file:

- processXML.py

5.9 Segmentation.Segmentation Class Reference

Public Member Functions

- def __init__
- def __str__ (self)
- def from_str (self, string)
- def pair_recall (self, truth)
- def pair_precision (self, truth)
- def pair_fscore (self, truth)
- def jaccard_precision
- def jaccard_recall
- def jaccard_fscore

Public Attributes

- segs

5.9.1 Detailed Description

```
A Segmentation is a list of Polygons, presumed pairwise non-overlapping.
```

5.9.2 Constructor & Destructor Documentation

5.9.2.1 def Segmentation.Segmentation.__init__ (*self, segments* = [], *string* = None, *fname* = None)

5.9.3 Member Function Documentation

5.9.3.1 def Segmentation.Segmentation.__str__ (*self*)

5.9.3.2 def Segmentation.Segmentation.from_str (*self, string*)

5.9.3.3 def Segmentation.Segmentation.jaccard_fscore (*self, truth, gamma* = 1.0)

```
the higher the gamma, the more perfection is prized
```

5.9.3.4 def Segmentation.Segmentation.jaccard_precision (*self, truth, gamma* = 1.0)

5.9.3.5 def Segmentation.Segmentation.jaccard_recall (*self, truth, gamma* = 1.0)

5.9.3.6 def Segmentation.Segmentation.pair_fscore (*self, truth*)

5.9.3.7 def Segmentation.Segmentation.pair_precision (*self, truth*)

5.9.3.8 def Segmentation.Segmentation.pair_recall (*self, truth*)

```
Returns probability that two random points from truth's segments
   will be classified the same way by truth and self (as belonging
   either to the same or to different articles)
```

5.9.4 Member Data Documentation

5.9.4.1 Segmentation.Segmentation.segs

The documentation for this class was generated from the following file:

- Segmentation.py

Chapter 6

File Documentation

6.1 Box.py File Reference

Classes

- class Box.Box

Namespaces

- Box

6.2 end2end.py File Reference

Classes

- class end2end.EndToEnd

Namespaces

- end2end

Functions

- def end2end.main ()

6.3 EvalOneToMany.py File Reference

Classes

- class EvalOneToMany.EvalOneToMany

Namespaces

- EvalOneToMany

Functions

- def EvalOneToMany.main ()

Variables

- int EvalOneToMany.NUM_CLASS = 3
- float EvalOneToMany.ONETOONE_THRESHOLD = 0.85
- float EvalOneToMany.ONETOMANY_THRESHOLD = 0.1
- list EvalOneToMany.LABELS = ['text', 'title', 'other']

6.4 Evaltest.py File Reference

Classes

- class Evaltest.EvalOneToMany

Namespaces

- Evaltest

Functions

- def Evaltest.main ()

Variables

- int Evaltest.NUM_CLASS = 3
- float Evaltest.ONETOONE_THRESHOLD = 0.85
- float Evaltest.ONETOMANY_THRESHOLD = 0.1
- list Evaltest.LABELS = ['text', 'title', 'other']

6.5 IdentityMetric.py File Reference

Classes

- class IdentityMetric.EvalOneToMany

Namespaces

- IdentityMetric

Functions

- def IdentityMetric.main ()

Variables

- int IdentityMetric.NUM_CLASS = 3
- float IdentityMetric.ONETOONE_THRESHOLD = 0.85
- float IdentityMetric.ONETOMANY_THRESHOLD = 0.1
- list IdentityMetric.LABELS = ['text', 'title', 'other']

6.6 Image.py File Reference

Classes

- class Image.NewsImage

Namespaces

- Image

6.7 imageBS.py File Reference

Namespaces

- imageBS

Variables

- list imageBS.f_image = sys.argv[1]
- list imageBS.f_xml = sys.argv[2]
- list imageBS.f_out = sys.argv[3]
- tuple imageBS.f = f_image.split('/')
- string imageBS.f_csv = 'images/'
- list imageBS.boxes = [ ]
- tuple imageBS.dat = line.strip('\n')
- dictionary imageBS.seg = {"annotations":[ ]}
- int imageBS.id = 0
- dictionary imageBS.tbInfo = {}

6.8 Latex.py File Reference

Namespaces

- Latex

Functions

- def Latex.replace_dict (string, replacements)
- def Latex.gen_latex (title, fnameout)
- def Latex.gen_pdf (fnametex, fnamepdf)

Variables

- tuple Latex.latex_template = f.read()

6.9 pipe_to_jason.py File Reference

Namespaces

- pipe_to_jason

Variables

- list pipe_to_jason.polys = [l.split('|') for l in f.read().split('\n') if l]
- list pipe_to_jason.labpolys = [(p[0],eval('[%s]'%(','.join(p[1:])))) for p in polys]
- int pipe_to_jason.sh = 6351
- int pipe_to_jason.sw = 3960
- dictionary pipe_to_jason.json

6.10 Polygon.py File Reference

Classes

- class Polygon.Polygon

Namespaces

- Polygon

6.11 processXML.py File Reference

Classes

- class processXML.ProcessXML

Namespaces

- processXML

6.12 readJSON.py File Reference

Namespaces

- readJSON

Functions

- def readJSON.seg_from_json (fname)

6.13 Segmentation.py File Reference

Classes

- class Segmentation.Segmentation

Namespaces

- Segmentation

6.14 testseg.py File Reference

Namespaces

- testseg

Functions

- def testseg.size_of_image (imname)

Variables

- list testseg.f_image = sys.argv[1]
- list testseg.f_xml = sys.argv[2]
- list testseg.f_out = sys.argv[3]
- tuple testseg.pxml = processXML.ProcessXML(f_xml)
- tuple testseg.tbList = pxml.getTBData()
- dictionary testseg.seg = {"annotations":[ ]}
- int testseg.id = 0
- dictionary testseg.tbInfo = {}

6.15 textblocksBS.py File Reference

Namespaces

- textblocksBS

Functions

- def textblocksBS.size_of_image (imname)

Variables

- list textblocksBS.f_image = sys.argv[1]
- list textblocksBS.f_xml = sys.argv[2]
- list textblocksBS.f_out = sys.argv[3]
- tuple textblocksBS.pxml = processXML.ProcessXML(f_xml)
- tuple textblocksBS.tbList = pxml.getTBData()
- dictionary textblocksBS.seg = {"annotations":[ ]}
- int textblocksBS.id = 0
- dictionary textblocksBS.tbInfo = {}

Index