Text Extraction Pipeline Documentation

Author: Falk Böschen *Version 0.6 – July 2016*

This document describes the different modules of the pipeline and especially their inputs and outputs. The whole pipeline is based on the streams framework[1]. Thus, the communication between different modules is achieved via a Data-Item which is handed from one module to the next. This Data-Item contains all the data and stores them as key-value pairs. The key is a string that is used to index the associated data object which can be of any serializable type. The following module descriptions include the key-values pairs that are used by each module. When a data object that is generated by one module should be used by another one, then the key of the input of the second module must equal the key of the output of the first module. An example can be seen below:

```
<de.cau.cs.kd.processor.open.OpenImage _fileKey="file" imageKey_="image" />
<de.cau.cs.kd.processor.image.grey.ATSCLuminanceConversion _imageKey="image" imageKey_="imageGrey" />
<de.cau.cs.kd.processor.image.binary.OtsuHierarchical _imageKey="imageGrey" imageKey_="imageBinary" />
```

The key of the output (imageKey_) is equal to the key that is input to the next module (_imageKey). If the same key is used for different items, then the old item gets replaced. One can see the parameter naming convention that was used in the example above. The incoming parameters begin with an underscore while the outgoing parameters end with an underscore.

A complete configuration file of the pipeline has to have the following structure:

```
<container>
<stream id="data" class="XXX" … />

        <process input="data">

                <… />

                <… />

                <… />

        </process>

</container>
```

The whole process is wrapped by a <container> tag. First an input has to be specified by defining a stream, which class is one of the source module described below. It is followed by a <process> tag which has to have an input key that is equal to the id key of the stream module. Inside the <process> tag one can list the processing and sink modules which shall be used.

[1] Streams Framework: https://sfb876.de/streams/

SOURCE MODULES **4**

GUIFileChooser 4
LocalFileChooser 4

PROCESSING MODULES **5**

Open File Modules **5**
OpenImage 5
OpenPDF 5
Image Conversion Modules **6**
ColorQuantization 6
HSVtoRGB 6
RGBtoHSV 6
ATSCLuminanceConversion 7
NTSCLuminanceConversion 7
EqualConversion 7
ChannelSelect 7
Image Binarization Modules **8**
ThresholdFix 8
ThresholdDynamic 8
Histogram 8
Niblack 9
Otsu 9
OtsuCheckboard 9
OtsuSliding 10
OtsuHierarchical 10
Other Image Processing Modules **10**
ColorToBinaryImages 10
Edge 11
Invert 11
Mean 11
Median 12
ConnectedComponentAnalysis 12
Pivoting 12
ImagesFromCluster 13
FilterForFigures 13
Feature Modules **14**
FeaturesFromCC 14
ComputeAngle 14
Filter Modules **15**
Area 15
Dots 15

Size 16
Clustering Modules **16**
DBSCAN 16
GravityGrouping 17
Grouping 17
MorphClustering 17
MST 18
OMST 18
OMSTSubclustering 18
Single 19
Singleton 19
OCR Modules **20**
Ocropy 20
Tesseract 20
Text Processing Modules **21**
NormalizeWhitespaces 21
QuantitativeAssessment 21
RemoveSpecialCharacters 21
RemoveStrings 21
Structural and other Modules **22**
Condense 22
Parallelize 22
ImageView 22

SINK MODULES **23**

Console **23**
CreateCSV **23**
CreateJSON **23**

Source Modules

Source modules are used to generate new Data-Items to send along the pipeline. They are defined inside the class parameter of a <stream> tag. The following modules are all in the de.cau.cs.kd.source package and are used to generate Data-Items that contain a file object pointing to a file on the local filesystem.

GUIFileChooser

The **GUIFileChooser** module presents the user with a file chooser dialog which is used to open one file at a time for processing.

Parameter	Type	Description
id	String	The ID of the stream which is used by <process> (default: "data")
limit	Long	The maximum number of files to read (default: "-1" – unlimited)
fileKey_	String	The key under which the File object is placed (default: "file")
_type	String	The type of files the file chooser can open (default: "." - all files)
_log	Boolean	Activate logging functionality (default: "false" - deactivated)
_logfile	String	Path to where the log information will be stored (default: null)

LocalFileChooser

The **LocalFileChooser** module iterates over all files inside a directory and sends one file after the other through the pipeline if the file has the correct type. Recursive mode is possible.

Parameter	Type	Description
id	String	The ID of the stream which is used by <process> (default: "data")
limit	Long	The maximum number of files to read (default: "-1" – unlimited)
fileKey_	String	The key under which the File object is placed (default: "file")
_type	String	The type of files that will be accepted (default: "." - all files)
_input	String	Path to the directory where the files are (default: not specified)
_recursive	Boolean	Activate recursive crawling of directory (default: "false" - deactivated)
_log	Boolean	Activate logging functionality (default: "false" - deactivated)
_logfile	String	Path to where the log information will be stored (default: null)

Processing Modules

Open File Modules

The following modules are located in the de.cau.cs.kd.processor.open package.

OpenImage

The **OpenImage** module expects as input a File object which points to an image file that is loaded and stored in the Data-Item.

Parameter	Type	Description
_fileKey	String	The key under which the File object is stored (default: "file")
imageKey_	String	The key under which the image is placed as a Double[][][] array (default: "image")
_gzip	Boolean	Expect a {ppm|pgm|pbm}.gz file as input (default: "false")

OpenPDF

The **OpenPDF** module expects as input a File object which points to a PDF file and extracts all text and images from this PDF and stores it in the Data-Item.

Parameter	Type	Description
_fileKey	String	The key under which the File object is stored (default: "file")
imagesKey_	String	The key under which all images are placed as a List<Double[][][]> (default: not specified)
textKey_	String	The key under which the text of the PDF is stored (default: not specified)

Image Conversion Modules

The following modules are used to convert images into different color codings. A Double[][][] structure is used as image format, where the first array dimension is the width of the image, the second dimension is the height of the image, and the third dimension is the color channel.

ColorQuantization

This module reduces the number of colors in an image to a maximal predefined number of colors.
The **ColorQuantization** module is located in the de.cau.cs.kd.processor.image.color package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_num	Integer	The (maximal) number of colors to reduce to (default: "8")
_windowSize	Integer	Parameter for the mean shift algorithm, which defines the pixel neighborhood for computing the mean value (default: "7")
_maxDepth	Integer	Parameter for the median cut algorithm, that defines the maximal depth of the 3D kd tree (default: "10")
_threshold	Double	Parameter for the mean shift algorithm to define the distance between different colors (default: "25.0")

HSVtoRGB

This module takes the color channel of an image and converts the values into RGB space, based on the premise that the input color channel had HSV encoding.
The **HSVtoRGB** module is located in the de.cau.cs.kd.processor.image.color package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")

RGBtoHSV

This module takes the color channel of an image and converts the values into HSV space, based on the premise that the input color channel had RGB encoding.
The **RGBtoHSV** module is located in the de.cau.cs.kd.processor.image.color package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")

ATSCLuminanceConversion

This module converts a color image into a grey image and thus reducing the color channel dimension from 3 to 1. It uses the luminance formula: y =0.2126R + 0.7152G + 0.0722B.
The module is located in the de.cau.cs.kd.processor.image.grey package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")

NTSCLuminanceConversion

This module converts a color image into a grey image and thus reducing the color channel dimension from 3 to 1. It uses the luminance formula: y =0.299R + 0.587G + 0.114B.
The module is located in the de.cau.cs.kd.processor.image.grey package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")

EqualConversion

This module converts a color image into a grey image and thus reducing the color channel dimension from 3 to 1. It uses the luminance formula: y = (R + G + B) / 3.
The module is located in the de.cau.cs.kd.processor.image.grey package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")

ChannelSelect

This module converts a color image into a grey image and thus reducing the color channel dimension from 3 to 1. It selects the specified color channel and discards the other channels.
The module is located in the de.cau.cs.kd.processor.image.grey package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_channel	Integer	The channel to keep. (default: "0")

Image Binarization Modules

The following modules are located in the de.cau.cs.kd.processor.image.binary package.

ThresholdFix

This module converts a grey image into a binary image.
It uses a fixed threshold to decide whether a pixel is assigned with the value 0 or 1.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_threshold	Double	The threshold for binarization. Range: 0-255 (default: "128.0")

ThresholdDynamic

This module converts a grey image into a binary image.
It uses a dynamic threshold to decide whether a pixel is assigned with the value 0 or 1.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_threshold	Double	Maximal difference from the average in the local window (default: "5.0")
_windowSize	Integer	Window size (default: "3")

Histogram

This module converts a grey image into a binary image.
It computes the histogram on the grey values, finds the maximum and sets all values that are above the most common color (minus a threshold) to 1 or else to 0.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_threshold	Double	Threshold parameter (default: "20.0")

Niblack

This module converts a grey image into a binary image using Niblack's binarization method.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_variant	String	The variant of Niblacks method to use. Options: "default", "wolf", "sauvola", "khurshid", "feng"
_windowSize	Integer	Parameter defining the neighborhood (default: "42")
_k	Double	Parameter used by all methods except "feng"
_r	Double	Parameter used by "sauvola", "wolf", and "feng"
_a1	Double	Parameter used by "feng" (default: "0.15")
_k1	Double	Parameter used by "feng" (default: "0.20")
_k2	Double	Parameter used by "feng" (default: "0.03")

Otsu

This module converts a grey image into a binary image using Otsu's method.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")

OtsuCheckboard

This module converts a grey image into a binary image by subdiving the image in a grid like fashion and applying Otsu's method to each grid cell.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_gridSizeX	Integer	Number of grid cells in horizontal direction (default: "5")
_gridSizeY	Integer	Number of grid cells in vertical direction (default: "5")

OtsuSliding

This module converts a grey image into a binary image by sliding a window over the image and computing Otsu's method in this window for each pixel.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_windowSizeX	Integer	Width of the sliding window
_windowSizeY	Integer	Height of the sliding window

OtsuHierarchical

This module converts a grey image into a binary image. It hierarchically subdivides the image where necessary and computes multiple thresholds using Otsu's method to binarize using the edge image.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_edgeBinarizationThreshold	Double	Threshold for binarizing the edge image (default: "32")
_minDimensionSize	Integer	Minimum Width/Height of a region to limit the subdivision (default: "15")
_hausdorffThreshold	Integer	Threshold for deciding whether to subdivide or not based on the computation of Hausdorff (default: "3")

Other Image Processing Modules

ColorToBinaryImages

This module takes a color image as input and creates a list of binary images – one for each color that is present in the image, where only the pixels are set to 1 where the original image has that color. The module is located in the de.cau.cs.kd.processor.image.binary package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
binaryImagesKey_	String	The key to put the List<Double[][][]> of binary images (default: "binaryImages")

Edge

Computes the edge image of an image.

The module is located in the de.cau.cs.kd.processor.image.operators package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the edge image is placed as a Double[][][] array (default: "image")
angleImageKey_	String	The key under which the angle image is placed as a Double[][][] array (default: "imageAngle")
gxImageKey_	String	The key under which the horizontal edge image is placed as a Double[][][] array (default: "Gx")
gyImageKey_	String	The key under which the vertical edge image is placed as a Double[][][] array (default: "Gy")
_method	String	The method to compute the edge image. Options: "sobel" (default), "roberts", "laplace1", and "laplace2"

Invert

This module inverts the color values of an RGB image.

The module is located in the de.cau.cs.kd.processor.image.operators package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")

Mean

This module filters an image with a mean filter, thus replacing each pixel with the average/mean value computed in a local window.

The module is located in the de.cau.cs.kd.processor.image.operators package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_windowSize	Integer	Size of the window (default: "3")

Median

This module filters an image with a mean filter, thus replacing each pixel with the median value in a local window. The module is located in the de.cau.cs.kd.processor.image.operators package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
imageKey_	String	The key under which the modified image is placed as a Double[][][] array (default: "image")
_windowSize	Integer	Size of the window (default: "3")

ConnectedComponentAnalysis

This module performs a connected component analysis on a binary image and outputs a list of regions, where each region is a list of pixel coordinates.
The module is located in the de.cau.cs.kd.processor.image.label package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
regionsKey_	String	The key for the list<list<pair<int,int>>> of regions (default: "regions")

Pivoting

This module creates a list of regions by applying a pivoting algorithm on the edge image which alternately computes the histogram projection of the x or y dimension and splitting the image based on the threshold value at the minimal points.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
_edgeImageKey	String	The key to access the edge image (Double[][][]) (default: "imageEdge")
regionsKey_	String	The key for the list<list<pair<int,int>>> of regions (default: "regions")
_threshold	Double	Threshold for subdividing (default : "20.0")
_maxDepth	Integer	The maximal subdivision depth (default: unlimited)

ImagesFromCluster

This module cuts out smaller images from an image which are defined by clusters of regions and stores them as individual Data-Items in a list of Data-Items.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
itemsKey_	String	The key under which the list of Data-Items is placed (default: "images")
imageKey_	String	The key under which the subimages are placed in their Data-Items (default: "subimage")
_clusterKey	String	The key for the cluster information list<list<int>> (default: "cluster")
_regionsKey	String	The key to the list<list<pair<int,int>>> of regions (default: "regions")
_anglesKey	String	The key for the angles list<Double> (default: "angles")
_border	Integer	Parameter defining the width of a border to add to the generated images (default: "5" pixel)

FilterForFigures

Filters a list of images based on the specified parameters. This module is located in the package de.cau.cs.kd.processor.image.classify.

Parameter	Type	Description
_imageListKey	String	The key under which the list of images (List<Double[][][]>) is found(default: "imagelist")
imageListKey_	String	The key under which the filtered list of images (List<Double[][][]>) is placed (default: "imagelist")
_minWidth	Integer	The minimum width in pixel an image must have (default: "500")
_minHeight	Integer	The minimum height in pixel an image must have (default: "500")
_maxWidth	Integer	The maximum width in pixel an image must have (default: "2000")
_maxHeight	Integer	The maximum height in pixel an image must have (default: "2000")
_minNumChar	Integer	The min. number of characters that have to be recognized using simple Tesseract OCR (default: "400")
_filterWidth	Boolean	Filter images by width if true (default: "true")
_filterHeight	Boolean	Filter images by height if true (default: "true")
_filterOCR	Boolean	Filter images by _minNumChar if true (default: "true")

Feature Modules

These modules compute numerical values based on other information.
They are located inside the de.cau.cs.kd.processor.features package.

FeaturesFromCC

Computes a list of feature vectors where each entry corresponds to a region and each feature vector consists of width and height of the region, x and y coordinate of its center of mass and its area occupation ratio.

Parameter	Type	Description
_regionsKey	String	The key to the list<list<pair<int,int>>> of regions (default: "regions")
featuresKey_	String	The key to the list<Vector> (default: "features")

ComputeAngle

Computes the orientation of text lines, which are defined as clusters of regions which are associated with features. Three different methods are available.

Parameter	Type	Description
_clusterKey	String	The key for the cluster information list<list<int>> (default: "cluster")
_featuresKey	String	The key for the features list<Vector> (default: "features")
_regionsKey	String	The key for the regions list<list<pair<int,int>>> (default: "regions")
anglesKey_	String	The key for the computed angles list<Double> (default: "angles")
_method	String	The method to compute the orientation. Options are: "hough", "ssod", and "psd" (default: "hough")

Filter Modules

The **Filter Modules** are used to filter a list of regions list<list<pair<int,int>>> with their features list<Vector> using certain thresholds. They are located in the de.cau.cs.kd.processor.filter package.

Area

This module filters regions based on the area (of their bounding box) they occupy.

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
_regionsKey	String	The key for the regions list<list<pair<int,int>>> (default: "regions")
featuresKey_	String	The key for the modified features list<Vector> (default: "features")
regionsKey_	String	The key for the modified regions list<list<pair<int,int>>> (default: "regions")
_upperratio	Double	Upper limit for the area occupation ratio (default: "1.0" – max)
_lowerratio	Double	Lower limit for the area occupation ratio (default: "0.0" – min)
_min	Integer	Minimum number of pixel for the region
_max	Integer	Maximum number of pixel for the region

Dots

This module assigns small regions that are most likely dots to the closest region that is not a dot.

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
_regionsKey	String	The key for the regions list<list<pair<int,int>>> (default: "regions")
featuresKey_	String	The key for the modified features list<Vector> (default: "features")
regionsKey_	String	The key for the modified regions list<list<pair<int,int>>> (default: "regions")
_merge	Boolean	If set to false, small regions are removed and not merged (default: "true")
_size	Integer	Maximum size(width or height) of a dot in pixel (default: "15")

Size

This module filters regions based on their width and/or height.

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
_regionsKey	String	The key for the regions list<list<pair<int,int>>> (default: "regions")
featuresKey_	String	The key for the modified features list<Vector> (default: "features")
regionsKey_	String	The key for the modified regions list<list<pair<int,int>>> (default: "regions")
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
_widthfactor	Double	Factor to multiply the standard deviation of the width distribution to define a range around the average width (default: "3.0")
_heightfactor	Double	Factor to multiply the standard deviation of the height distribution to define a range around the average height (default: "3.0")
_relativeMax	Double	Maximum width/height defined by a relative factor to the image width/height (default: "1.0")
_relativeMin	Double	Minimum width/height defined by a relative factor to the image width/height (default: "0.0")

Clustering Modules

The following modules are located in the de.cau.cs.kd.processor.clustering package.

DBSCAN

This module creates a clustering on regions based on their feature vectors.

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
clusterKey_	String	The key for the cluster information list<list<int>> (default: "cluster")
_adaptive	String	List of indices of feature vector entries to use for computing the _eps parameter adaptively [optional]
_eps	Double	Parameter defining the neighborhood of a region (default: "15.0")
_minPts	Integer	The minimum number of points required to start a cluster (default: "1")
_separateNoise	Boolean	If "true", a separate cluster for all regions that are classified as noise is created. Otherwise the noise objects are discarded. (default: "false")

GravityGrouping

This module groups/clusters regions using a method proposed by Weihua Huang et al.[2] which is based on Newton's Formula for gravity.

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
_regionsKey	String	The key for the regions list<list<pair<int,int>>> (default: "regions")
clusterKey_	String	The key for the cluster information list<list<int>> (default: "cluster")
_threshold	Double	Threshold for the distance that is computed by the gravity formula. (default: "20.0")

Grouping

This module groups regions using another method proposed by Weihua Huang and C.L. Tan.[3]

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
_regionsKey	String	The key for the regions list<list<pair<int,int>>> (default: "regions")
clusterKey_	String	The key for the cluster information list<list<int>> (default: "cluster")
_k	Double	Parameter in the grouping function (default: "10.0")

MorphClustering

A morphological clustering based on the work of Chiang and Knoblock.[4]

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
_regionsKey	String	The key for the regions list<list<pair<int,int>>> (default: "regions")
clusterKey_	String	The key for the cluster information list<list<int>> (default: "cluster")
_maxCurvatureRatio	Double	Parameter for the morphological clustering (default: "1.2")
_maxSizeRatio	Double	Parameter for the morphological clustering (default: "2.0")
_maxDistanceRatio	Double	Parameter for the morphological clustering (default: "2.0")

[2] Huang, W.; Tan, C. L. & Leow, W. K. Associating Text and Graphics for Scientific Chart Understanding Eighth International Conference on Document Analysis and Recognition (ICDAR 2005), 29 August - 1 September 2005, Seoul, Korea, IEEE Computer Society, 2005, 580-584

[3] Huang, W. & Tan, C. L. King, P. R. & Simske, S. J. (Eds.) A system for understanding imaged infographics and its applications Proceedings of the 2007 ACM Symposium on Document Engineering, Winnipeg, Manitoba, Canada, August 28-31, 2007, ACM, 2007, 9-18

[4] Chiang, Y. & Knoblock, C. A. Recognizing text in raster maps GeoInformatica, 2015, 19, 1-27

MST

This module applies a minimum spanning tree clustering onto the regions. The MST can be split using one of two methods: Threshold-based, or Inconsistency-based.

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
clusterKey_	String	The key for the cluster information list<list<int>> (default: "cluster")
_threshold	Double	A length threshold for removing edges
_inconsistency	Boolean	If set to "true", the average edge length at a node is computed and all edges longer than the average times 1.2 are removed.

OMST

This module subdivides an existing clustering using a minimum spanning tree approach that is split using orientation information.

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
_clusterKey	String	The key for the cluster information list<list<int>> (default: "cluster")
clusterKey_	String	The key for the new cluster information list<list<int>> (default: "cluster")
_angle	String	Two comma separated angles in degree that define the range around the mean orientation of a cluster where all edges with orientations outside this range are removed. (default: "60,60")

OMSTSubclustering

A slightly different version of the implementation of the **OMST** module.

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
_clusterKey	String	The key for the cluster information list<list<int>> (default: "cluster")
clusterKey_	String	The key for the new cluster information list<list<int>> (default: "cluster")
_angle	String	Two comma separated angles in degree that define the range around the mean orientation of a cluster where all edges with orientations outside this range are removed. (default: "60,60")

Single

This module creates a single cluster containing all regions.

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
clusterKey_	String	The key for the cluster information list<list<int>> (default: "cluster")

Singleton

This module creates an individual cluster for every region.

Parameter	Type	Description
_featuresKey	String	The key for the features list<Vector> (default: "features")
clusterKey_	String	The key for the cluster information list<list<int>> (default: "cluster")

OCR Modules

The following modules are located in the de.cau.cs.kd.processor.ocr package.

Ocropy

This module is accessing the Ocropy installation on a Unix system via command line to recognize the text inside an image and returning the recognized text as a String.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
textKey_	String	The key to store the recognized text (default: "ocredtext")
_tmpfolder	String	The folder which the Ocropy installation shall use for its temporary data (default: "book")
_option	String	The parameters that shall be passed to Ocropy (default: none)

Tesseract

This module uses the Tess4J API to apply Tesseract OCR on the input image. Output is a text String.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
textKey_	String	The key to store the recognized text (default: "ocredtext")
_language	String	The language model Tesseract shall use (default: "eng") (based on available dataset)
_OEMmode	String	Set the OEM mode[5] of tesseract (default: "OEM_TESSERACT_ONLY")
_PSMmode	String	Set the PSM mode[6] of tesseract (default: " PSM_OSD_ONLY")

[5] Options are:
"OEM_CUBE_ONLY", "OEM_DEFAULT", "OEM_TESSERACT_ONLY", "OEM_TESSERACT_CUBE_COMBINED".
See http://tess4j.sourceforge.net/docs/docs-3.0/net/sourceforge/tess4j/ITessAPI.TessOcrEngineMode.html

[6] See http://tess4j.sourceforge.net/docs/docs-3.0/net/sourceforge/tess4j/ITessAPI.TessPageSegMode.html

Text Processing Modules

These modules can be used to process text strings.

The following modules are located in the de.cau.cs.kd.processor.text package.

NormalizeWhitespaces

This module replaces all whitespaces (even sequences) with single blank spaces.

Parameter	Type	Description
_textKey	String	The key to the String to process (default: "ocredtext")
textKey_	String	The key where the processed String is stored (default: "processedtext")

QuantitativeAssessment

Sets a String to null if its number of characters divided by the number of regions that were used to create these characters is greater than the _maxRatioDifference.

Parameter	Type	Description
_textKey	String	The key to the String to process (default: "ocredtext")
textKey_	String	The key where the processed String is stored (default: "processedtext")
_maxRatioDifference	Double	The maximal ration between characters and regions (default: "0.1")

RemoveSpecialCharacters

Removes special characters from a String.

Parameter	Type	Description
_textKey	String	The key to the String to process (default: "ocredtext")
textKey_	String	The key where the processed String is stored (default: "processedtext")
_remove	String	A string of the characters to removed (default: all special characters)

RemoveStrings

Sets text strings that contain more than _num special characters to null.

Parameter	Type	Description
_textKey	String	The key to the String to process (default: "ocredtext")
textKey_	String	The key where the processed String is stored (default: "processedtext")
_remove	String	A string of the characters which are detected to decide whether to remove a String (default: all special characters)
_num	Integer	The max. allowed number of special characters

Structural and other Modules

Condense

This module merges a list of multiple Data-Items into one Data-Item.
This module is located inside the de.cau.cs.kd.processor package.

Parameter	Type	Description
_itemsKey	String	The key to the Data-Item list (default: none)
_mergeKey	String	A list of comma separated keys which shall be merged (default: none)
_append	Boolean	Merges attributes that are list by extending the list and not creating a list over lists. (default: "true")
_keepNull	Boolean	If set to false, null attributes are skipped (default: "true")

Parallelize

This module splits a Data-Item into multiple Data-Items for parallel processing while splitting an attribute and duplicating the rest. This module is located inside the de.cau.cs.kd.processor package.

Parameter	Type	Description
_splitKey	String	The key of the list or array attribute to split (default: none)
itemsKey_	String	The key where the list of newly created Data-Items are stored (default: none)

ImageView

This module shows an image with the option to additionally visualize regions or cluster.
This module is located inside the de.cau.cs.kd.processor.view package.

Parameter	Type	Description
_imageKey	String	The key to access the image (Double[][][]) (default: "image")
_regionsKey	String	The key to access the regions (default: null) [optional]
_featuresKey	String	The key to access features (default: null) [optional] (required for regions and cluster visualization)
_clusterKey	String	The key to access the clusters (default: null) [optional]
_file	String	The path to where the image shall be saved (default: none) [optional]

Sink Modules

The Sink Modules are designed to output the content of the current Data-Item. They are not real sinks that terminate the execution, but a different kind of processing modules. Thus, they need to be inside the <process> tags as well. They are located inside the de.cau.cs.kd.sink package.

Console

The **Console** module prints all contents of the Data-Item to the standard output, except for nested lists, which are skipped. It outputs each key together with its data. No parameters are used.

Parameter	Type	Description
-	-	-

CreateCSV

The **CreateCSV** module generates a file where each line represents an object which is described by the values in the different columns which are separated by a pre-defined symbol. It requires one or multiple lists as input where the i^{th} item in one list describes the same object as the i^{th} item in any of the other lists.

Parameter	Type	Description
_file	String	Path to where the CSV file will be stored (default: not specified) [optional] (required if _fileKey not set)
_fileKey	String	The key under which the File object is located (default: null) [optional] (required if _file not set)
_separator	String	The separating character that is used to separate the different columns in the generated file (default: ”,”) [optional]
_itemsKey	String	A comma separated list of keys that all point to list object containing the values to output (default: null)
_suffix	String	A suffix that is added to the generated files name. (default: none) [optional]
_prefix	String	A prefix that is added to the generated files name. (default: none) [optional]

CreateJSON

The **CreateJSON** module works similar like the **CreateCSV** module but instead of a CSV file, it creates a JSON file with additional meta-information.

Parameter	Type	Description
_fileKey	String	The key under which the File object is located (default: null) for meta-data extraction
_imageKey	String	The key under which the image data is located (default: null)
_itemsKey	String	A comma separated list of keys that all point to list object containing the values to output (default: null)
_outputFolder	String	Path to the directory where the JSON file(s) will be stored (default: not specified) [optional]