AFS

Generated by Doxygen 1.8.6

Fri Mar 6 2015 11:22:10

Contents

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

Module Documentation

4.1 CUDA_PRIMALDUAL_OPTIMIZATION

Functions

- __global__ void kernel_binarize_u (float ∗u, unsigned char ∗u_binary, int width, int height, int n, int pitch, int pitchUChar)

  CUDA kernel function to binarize the relaxed solution u.
- void call_binarization (float ∗u, unsigned char ∗u_binary, int width, int height, int n)

  call function to binarize the solution on CUDA
- void __cudaSafeCall (cudaError err, const char ∗file, const int line)

  call function to print out CUDA errors
- __global__ void kernel_grad_ascent (float ∗u_bar, float ∗xi, float ∗psi, float ∗sum_u, float ∗g, int width, int height, int n, float lambda, int pitch)

  CUDA kernel function for gradient ascent.
- __global__ void kernel_grad_descent (float ∗dataterm, float ∗u, float ∗u_bar, float ∗xi, float ∗psi, float ∗sum-_u, int width, int height, int n, int pitch)

  CUDA kernel function for gradient descent.
- void call_segmentation (float ∗dataterm, float ∗g, float ∗u, int width, int height, int n, float lambda, int &max-Steps, double &time_segm)

  call function to run segmentation on gpu

4.1.1 Detailed Description

This group consists of CUDA kernel and c-call functions for primal-dual algorithm

4.1.2 Function Documentation

4.1.2.1 void __cudaSafeCall (cudaError *err,* const char ∗ *file,* const int *line*) `[inline]`

call function to print out CUDA errors

Parameters

err	CUDA error

file	output file
line	output line

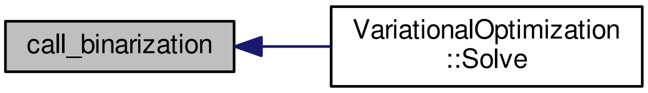
4.1.2.2 void call_binarization (float ∗ *u,* unsigned char ∗ *u_binary,* int *width,* int *height,* int *n*)

call function to binarize the solution on CUDA

Parameters

u	relaxed optimized solution
u_binary	binary solution
width	width of image
height	height of image
n	number of regions(classes)

Here is the caller graph for this function:



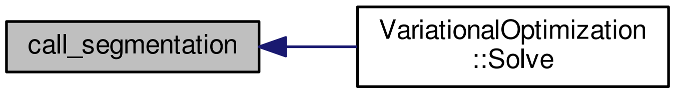
4.1.2.3 void call_segmentation (float ∗ *dataterm,* float ∗ *g,* float ∗ *u,* int *width,* int *height,* int *n,* float *lambda,* int & *maxSteps,* double & *time_segm*)

call function to run segmentation on gpu

Parameters

dataterm	dataterm
g	edge detection function
u	region indicator variable
width	width of the image
height	height of the image
n	number of labels(classes)
lambda	weighting parameter
maxSteps	(maximum) number of iterations
time_segm	measures the runtime

Here is the caller graph for this function:



4.1.2.4 __global__ void kernel_binarize_u (float ∗ *u,* unsigned char ∗ *u_binary,* int *width,* int *height,* int *n,* int *pitch,* int *pitchUChar*)

CUDA kernel function to binarize the relaxed solution u.

Parameters

u	relaxed optimized solution
u_binary	binary solution
width	width of image
height	height of image
n	number of regions(classes)
pitch	CUDA memory management
pitchUChar	CUDA memory management

4.1.2.5 __global__ void kernel_grad_ascent (float ∗ *u_bar,* float ∗ *xi,* float ∗ *psi,* float ∗ *sum_u,* float ∗ *g,* int *width,* int *height,* int *n,* float *lambda,* int *pitch*)

CUDA kernel function for gradient ascent.

Parameters

u_bar	overrelaxation of u
xi	dual variable
psi	Lagrange multiplier for simplex constraint
sum_u	variable for the simplex constraint
g	edge detection function
width	width of the image
height	height of the image
n	number of labels(classes)
lambda	weighting parameter
pitch	CUDA memory management

4.1.2.6 __global__ void kernel_grad_descent (float ∗ *dataterm,* float ∗ *u,* float ∗ *u_bar,* float ∗ *xi,* float ∗ *psi,* float ∗ *sum_u,* int *width,* int *height,* int *n,* int *pitch*)

CUDA kernel function for gradient descent.

Parameters

dataterm	dataterm
u	region indicator variable
u_bar	overrelaxation of u
xi	dual variable
psi	Lagrange multiplier for simplex constraint
sum_u	variable for the simplex constraint
width	width of the image
height	height of the image
n	number of labels(classes)
pitch	CUDA memory management

Chapter 5

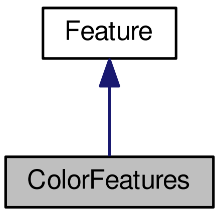
Class Documentation

5.1 ColorFeatures Class Reference

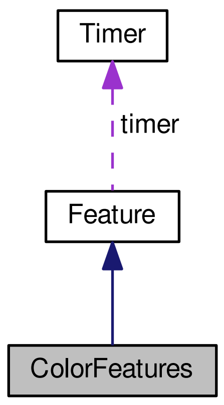
The ColorFeatures class.

`#include <Feature.hpp>`

Inheritance diagram for ColorFeatures:



Collaboration diagram for ColorFeatures:



Public Member Functions

- ColorFeatures (vector< Size > &all_patches, string colorFolder, string ext, int numSubSample)

  Class Constructor.
- void extractFeatures (Image ∗im, Mat &features)

  Extract features.

Public Attributes

- vector< Point2d > rndLocs

  relative pixel positions
- vector< Point2d > rndRelatives

  relative pixel positions
- vector< Rect > rndPatches

  relative patch positions

Additional Inherited Members

5.1.1 Detailed Description

The ColorFeatures class.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 ColorFeatures::ColorFeatures (vector< Size > & *all_patches*, string *colorFolder*, string *ext*, int *numSubSample*)

Class Constructor.

Parameters

all_patches	set of all patches
colorFolder	folder to store features
ext	extension of feature files
numSubSample	sub sample size

5.1.3 Member Function Documentation

5.1.3.1 void ColorFeatures::extractFeatures (Image ∗ *im*, Mat & *features*) [virtual]

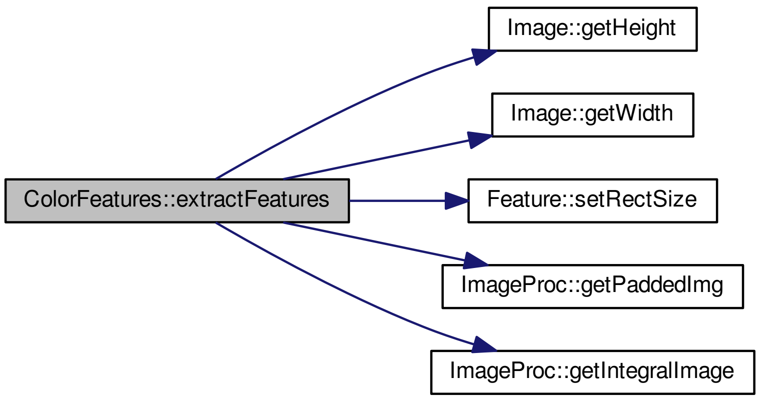
Extract features.

Parameters

im	image
features	feature matrix

Implements Feature.

Here is the call graph for this function:



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

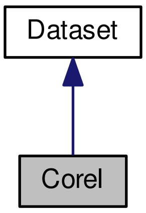
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Feature.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Feature.cpp

5.2 Corel Class Reference

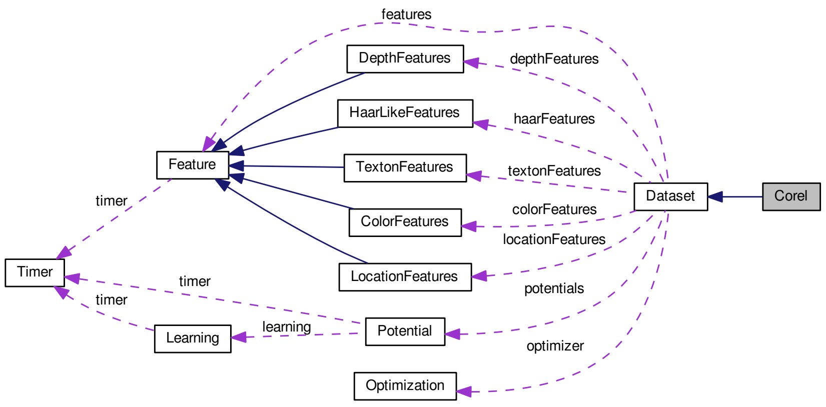
The Corel class.

`#include <Dataset.hpp>`

Inheritance diagram for Corel:



Collaboration diagram for Corel:



Public Member Functions

- Corel ()

  Class Constructor.
- void RGB2Label (Mat rgb, Mat &labels)

  Convert from RGB to Label.
- void Label2RGB (Mat labels, Mat &rgb)

  Convert from label to RGB.

Additional Inherited Members

5.2.1 Detailed Description

The Corel class.

5.2.2 Member Function Documentation

5.2.2.1 void CoreI::Label2RGB (Mat *labels*, Mat & *rgb*) `[virtual]`

Convert from label to RGB.

Parameters

labels	label matrix
rgb	RGB image

Reimplemented from Dataset.

5.2.2.2 void CoreI::RGB2Label (Mat *rgb*, Mat & *labels*) [virtual]

Convert from RGB to Label.

Parameters

rgb	RGB image
labels	label matrix to store labels for each pixel

Reimplemented from Dataset.

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

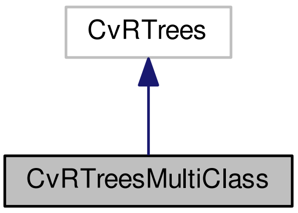
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Dataset.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Dataset.cpp

5.3 CvRTreesMultiClass Class Reference

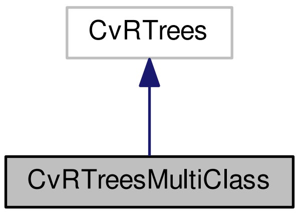
The CvRTreesMultiClass class This class has been taken from the following link: `http://stackoverflow.-com/questions/10358964/using-opencv-random-forests-is-there-any-way-to-obtain-the-confic` This function estimates the class probabilities given a feature vector. Each tree votes for one class label.

`#include <Learning.hpp>`

Inheritance diagram for CvRTreesMultiClass:



Collaboration diagram for CvRTreesMultiClass:



Public Member Functions

- int predict_multi_class (const CvMat ∗sample, int out_votes[], const CvMat ∗missing=0) const

    Predict unnormalized class probability for one test sample.

5.3.1 Detailed Description

The CvRTreesMultiClass class This class has been taken from the following link: `http://stackoverflow.-com/questions/10358964/using-opencv-random-forests-is-there-any-way-to-obtain-the-confid`
This function estimates the class probabilities given a feature vector. Each tree votes for one class label.

5.3.2 Member Function Documentation

5.3.2.1 int CvRTreesMultiClass::predict_multi_class (const CvMat ∗ *sample*, int *out_votes[ ]*, const CvMat ∗ *missing* = 0) const [inline]

Predict unnormalized class probability for one test sample.

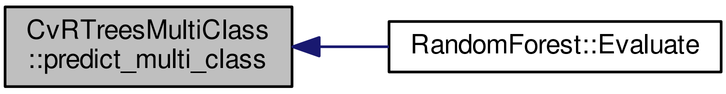
Parameters

sample	Test sample
out_votes	Unnormalized class probabilities for a given test sample
missing	Optional missing measurement mask of the sample (OpenCV doc)

Returns

Total number of trees in the forest

Here is the caller graph for this function:



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

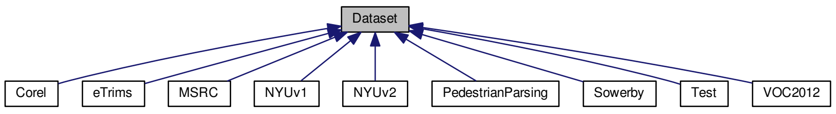
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Learning.hpp

5.4 Dataset Class Reference

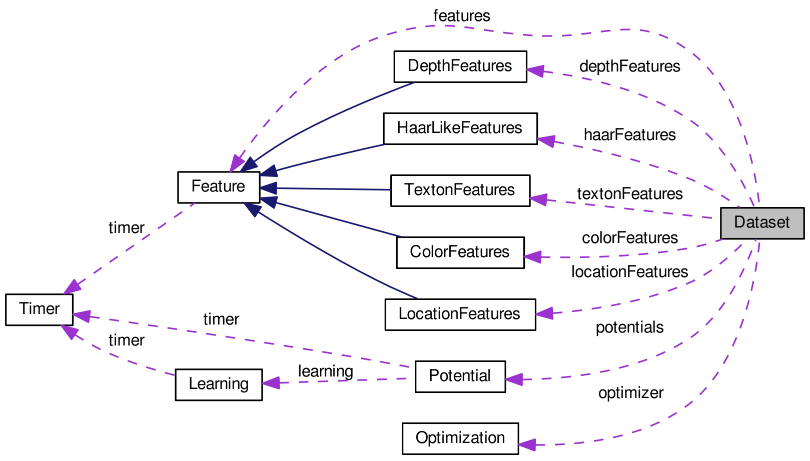
The Dataset class.

`#include <Dataset.hpp>`

Inheritance diagram for Dataset:



Collaboration diagram for Dataset:



Public Member Functions

- Dataset ()

  Class Constructor.
- virtual ∼Dataset ()

  Class Deconstructor.
- virtual void RGB2Label (Mat rgb, Mat &labels)

  Convert from RGB to Label.
- virtual void Label2RGB (Mat labels, Mat &rgb)

  Convert from label to RGB.
- virtual void getDatasetImages (int trainCount, int testCount)

  Split Dataset.

- virtual void readImgsInDir (vector< string > &imgs, string folder, string ext)

  Read Dataset.
- virtual bool readSplitsToVectors ()

  Read pre-splitted train/validation/test images.
- virtual void writeSplitsToFile ()

  Write train/validation/test splits to files.
- virtual int createDir (string dirName)

  Create directory.
- virtual int clearDir (string dirName)

  Remove all file in directory.
- virtual int findImage (string name)

  Find image index in testImgs.
- virtual void createResultFolders ()

  Create Folders for Results.
- virtual int getFullFeatureDim ()

  Get size of full feature set.
- virtual void removeUnlabeledImages ()

  Remove Unlabeled Images from Dataset.

Public Attributes

- vector< string > allImages

  list of all images in dataset
- vector< string > trainImgs

  list of training images
- vector< string > testImgs

  list of test images
- vector< string > validationImgs

  list of validation images
- vector< int > selectedFeatures

  vector stores indices of selected features
- bool isSplitDataset

  flag to split dataset randomly
- bool savePotentials

  flag to save potentials
- bool saveDetections

  flag to save detection results (argmax of potential for each pixel on image)
- bool showEntropy

  flag to visualize entropy map of the image
- int trainCount

  size of the training set
- int testCount

  size of the test test
- double trainEntropy

  entopy of the training data (balance of the classes in training set)
- double testEntropy

  entropy of the test set (how certain the detection is)
- string name_

  name of the dataset
- string resultDir

folder to store the segmentation outputs
- string trainFolder

  folder to store classifiers
- string resultFolder

  folder to store results. (e.g. mainFolder + '/Result')
- string mainFolder

  main folder of dataset
- string imageFolder

  folder where RGB image files are
- string imageFileExt

  extension of RGB image files
- string grFolder

  folder where ground truth files are
- string grFileExt

  extension of ground truth files
- string depthFolder

  folder to store depth features
- string depthFileExt

  extension of depth feature files
- string haarFolder

  folder to store haar-like features
- string haarFeatureExt

  extension of haar-like feature files
- string colorFolder

  folder to store color features
- string colorFeatureExt

  extension of color feature files
- string locationFolder

  folder to store location features
- string locationFeatureExt

  extension of location feature files
- string textonFolder

  folder to store texton features
- string textonFeatureExt

  extension of texton feature files
- string depthFeatureFolder

  folder to store depth features
- string depthFeatureExt

  extension of depth feature files
- string potentialFolder

  folder to stor potentials
- string potentialExt

  extension of potential files
- string detectionFolder

  folder to store detection results
- string pMapFolder

  folder to store potential maps
- int numFeatureDims

  total size of current feature set
- int fcount_

  Number of features in the current(selected) feature set.

- int n

  maximum number of regions (== numClasses)
- int maxSteps

  maximum iteration number for VariationalOptimization
- float lambda

  smoothing parameter lambda for VariationalOptimization
- int subSample

  sub sample size
- double textonBandWidth

  bandwidth of kernels for TextonFeautures
- unsigned int numClasses

  total number of classes in Dataset
- HaarLikeFeatures ∗ haarFeatures

  instance of HaarLikeFeatures
- ColorFeatures ∗ colorFeatures

  instance of ColorFeatures
- LocationFeatures ∗ locationFeatures

  instance of LocationFeatures
- TextonFeatures ∗ textonFeatures

  instance of TextonFeatures
- DepthFeatures ∗ depthFeatures

  instance of DepthFeatures
- const float ∗ class_weights

  Weights of each class to balance the training error.
- CvRTParams RFParams

  Structure to store parameteres for Random Forest.
- CvTermCriteria RFterm_crit

  termination criteria for Random Forests training
- int RFmax_num_of_trees_in_the_forest

  maximum number of trees in the forest
- float RFforest_accuracy

  sufficient accuracy, OOB error (OpenCV docs)
- int totalFeatureType

  total number of feature types
- int numPotentials

  number of potentails
- Feature ∗∗ features

  pointer to the list of features
- Potential ∗∗ potentials

  pointer to the list of potentials
- Optimization ∗ optimizer

  pointer to the optimizer

Protected Attributes

- int index

  variable to store index

5.4.1 Detailed Description

The Dataset class.

5.4.2 Member Function Documentation

5.4.2.1 int Dataset::clearDir (string *dirName*) `[virtual]`

Remove all file in directory.

Parameters

dirName	directory to be cleared

Returns

1 if all files succesfully removed

5.4.2.2 int Dataset::createDir (string *dirName*) `[virtual]`

Create directory.

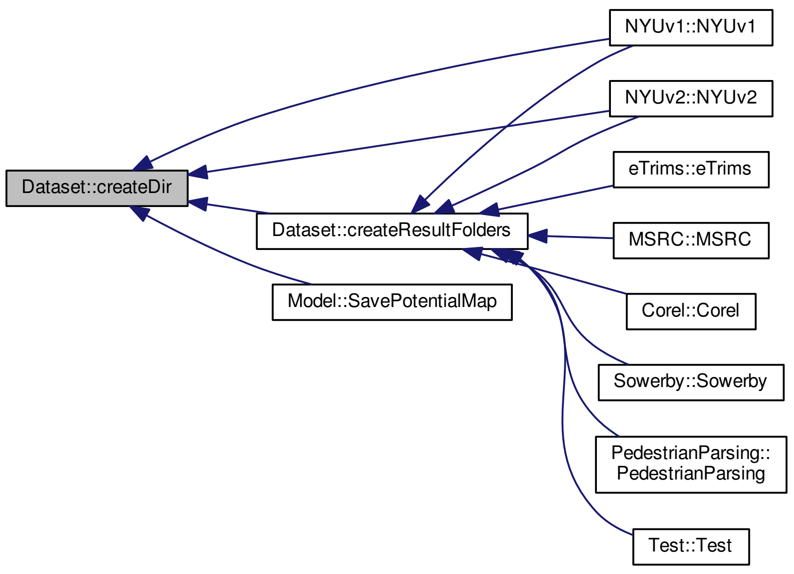
Parameters

dirName	name of directory

Returns

1 if directory exist

Here is the caller graph for this function:



5.4.2.3 int Dataset::findImage (string *name*) `[virtual]`

Find image index in testImgs.

Parameters

name	name of the image

Returns

index of the image in testImgs

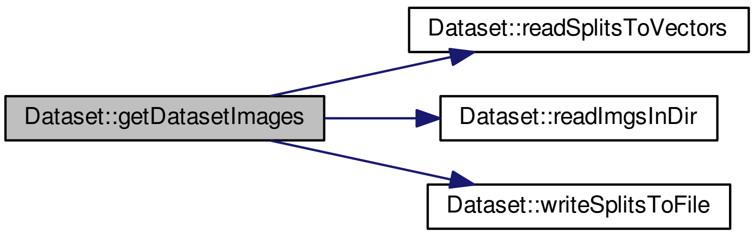
5.4.2.4 void Dataset::getDatasetImages (int *trainCount*, int *testCount*) [virtual]

Split Dataset.

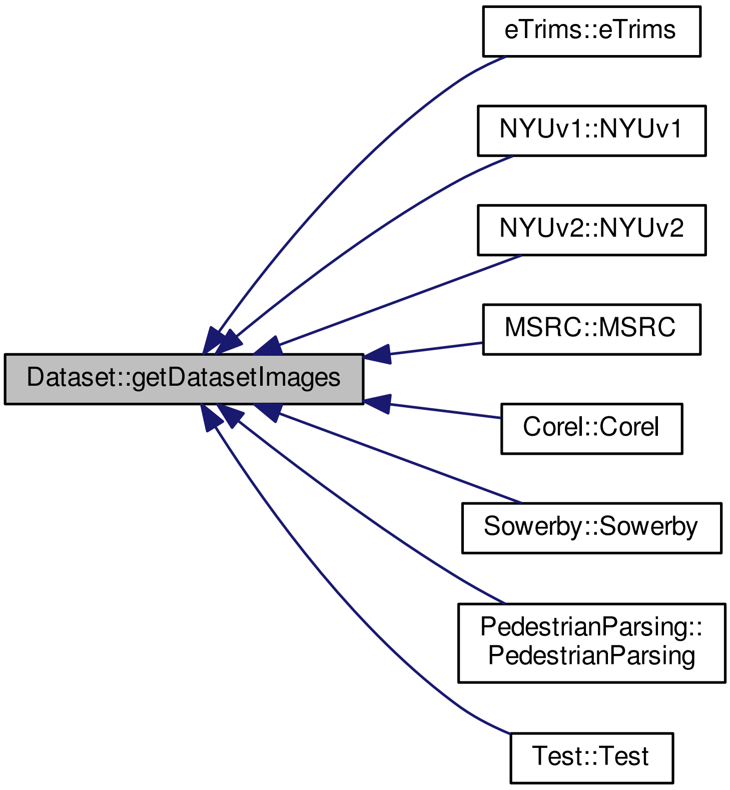
Parameters

trainCount	number of training samples, either a number or a proportion
testCount	number of test samples, either a number or a proportion

Here is the call graph for this function:


Dataset::getDatasetImages
Dataset::readSplitsToVectors
Dataset::readImgsInDir
Dataset::writeSplitsToFile

Here is the caller graph for this function:



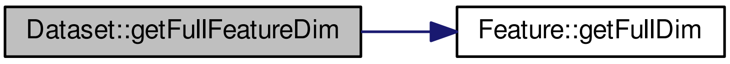
5.4.2.5 int Dataset::getFullFeatureDim ( ) [virtual]

Get size of full feature set.

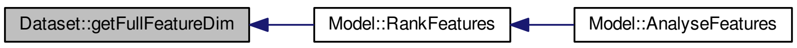
Returns

size of full feature set

Here is the call graph for this function:



Here is the caller graph for this function:



5.4.2.6 void Dataset::Label2RGB (Mat *labels*, Mat & *rgb*) `[virtual]`

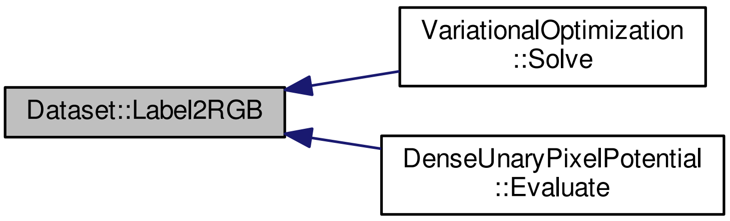
Convert from label to RGB.

Parameters

labels	label matrix
rgb	RGB image

Reimplemented in PedestrianParsing, VOC2012, NYUv2, NYUv1, Corel, MSRC, and eTrims.

Here is the caller graph for this function:


VariationalOptimization ::Solve
Dataset::Label2RGB
DenseUnaryPixelPotential ::Evaluate

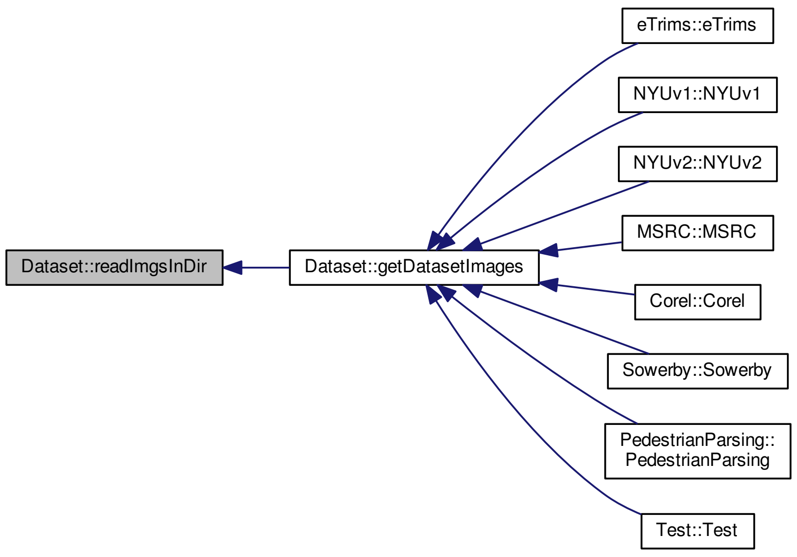
5.4.2.7 void Dataset::readImgsInDir (vector< string > & *imgs*, string *folder*, string *ext*) `[virtual]`

Read Dataset.

Parameters

imgs	vector of image names
folder	folder to read images from
ext	extension of image files

Here is the caller graph for this function:



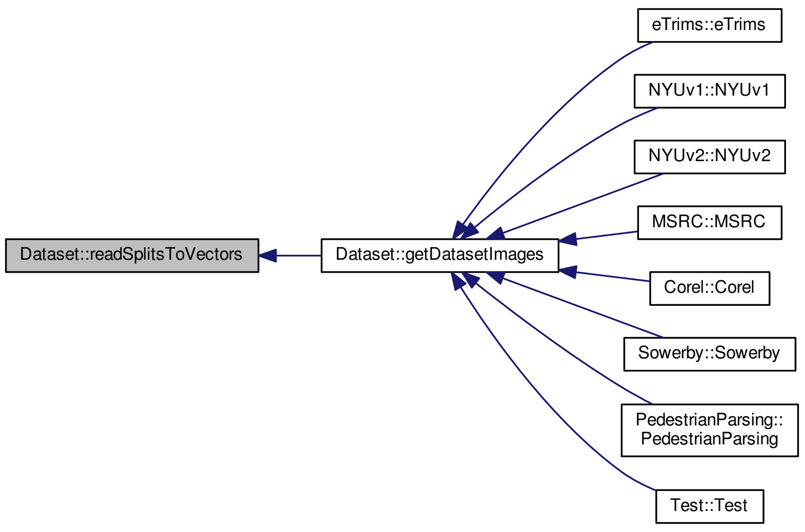
5.4.2.8 bool Dataset::readSplitsToVectors ( ) `[virtual]`

Read pre-splitted train/validation/test images.

Returns

true if split files exist

Here is the caller graph for this function:


eTrims::eTrims
NYUv1::NYUv1
NYUv2::NYUv2
MSRC::MSRC
Dataset::readSplitsToVectors
Dataset::getDatasetImages
Corel::Corel
Sowerby::Sowerby
PedestrianParsing::PedestrianParsing
Test::Test

5.4.2.9 void Dataset::RGB2Label (Mat *rgb*, Mat & *labels*) `[virtual]`

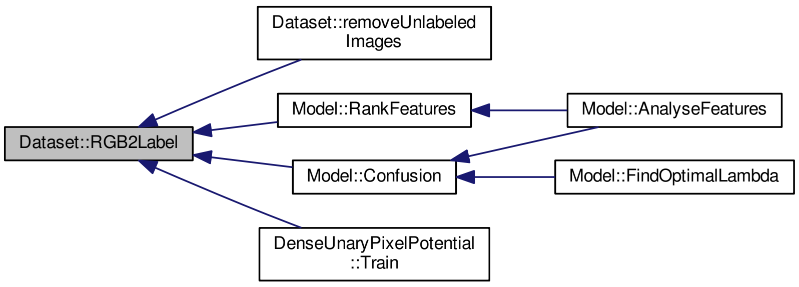
Convert from RGB to Label.

Parameters

rgb	RGB image
labels	label matrix to store labels for each pixel

Reimplemented in PedestrianParsing, VOC2012, NYUv2, NYUv1, Corel, MSRC, and eTrims.

Here is the caller graph for this function:



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

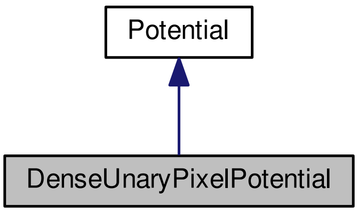
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Dataset.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Dataset.cpp

5.5 DenseUnaryPixelPotential Class Reference

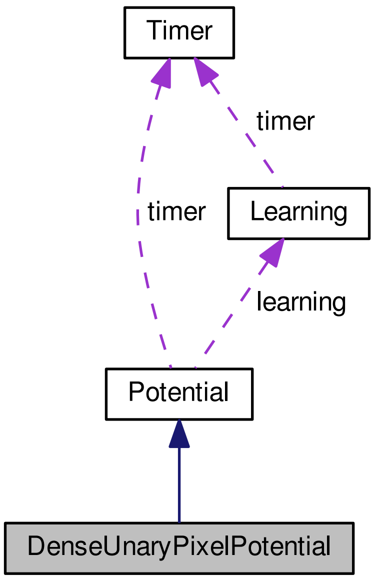
The DenseUnaryPixelPotential class.

`#include <Potential.hpp>`

Inheritance diagram for DenseUnaryPixelPotential:



Collaboration diagram for DenseUnaryPixelPotential:



Public Member Functions

- DenseUnaryPixelPotential (string dir, string fileExt)

    Class Constructior.
- int Train (Dataset ∗dataset, vector< string > &imageList, int from, int to, bool FAST_COMPUTATION=false)

    Train potential.
- int Evaluate (Dataset ∗dataset, vector< string > &imageList, int from, int to, bool FAST_COMPUTATIO-N=false)

    Evaluate potentials.

Additional Inherited Members

5.5.1 Detailed Description

The DenseUnaryPixelPotential class.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 DenseUnaryPixelPotential::DenseUnaryPixelPotential (string *dir,* string *fileExt*)

Class Constructior.

Parameters

dir	Output directory
fileExt	file extension

5.5.3 Member Function Documentation

5.5.3.1 int DenseUnaryPixelPotential::Evaluate (Dataset $*$ *dataset*, vector$<$ string $>$ & *imageList*, int *from*, int *to*, bool *FAST_COMPUTATION* = `false`) `[virtual]`

Evaluate potentials.

Parameters

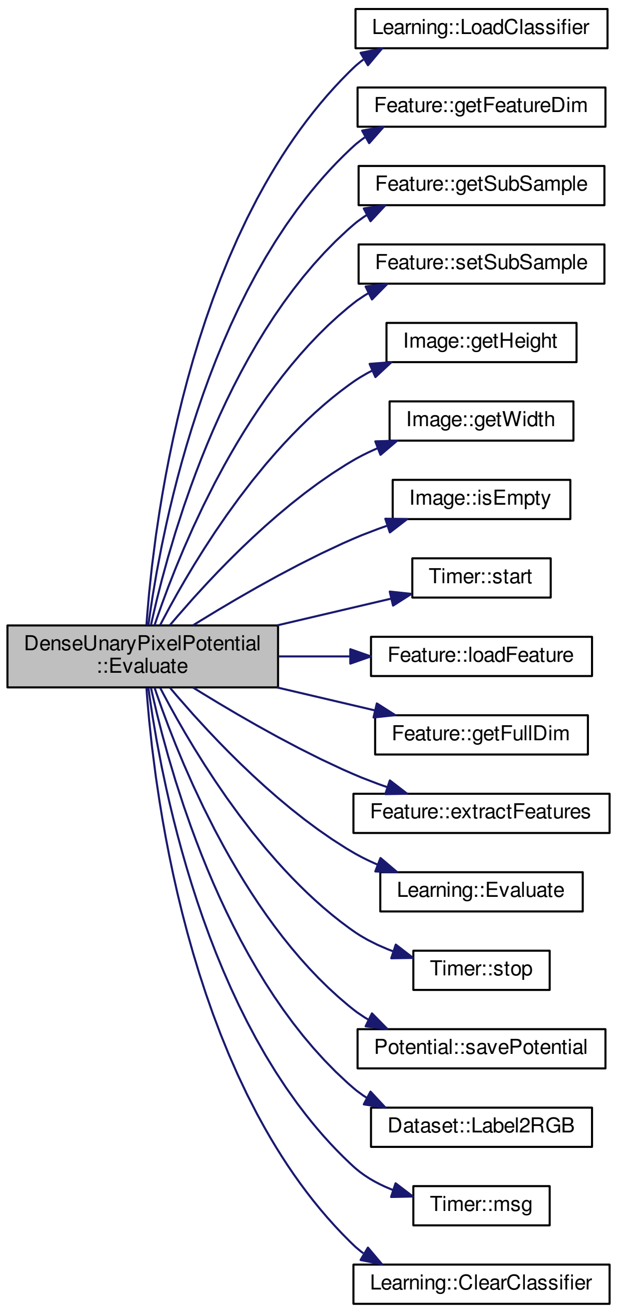
dataset	pointer to the current dataset
imageList	list of images
from	start index
to	stop index, set to -1 to process all images startin from "from"
FAST_COMPUTATION	flag to activate fast computation

Returns

estimated time per image in miliseconds

Implements Potential.

Here is the call graph for this function:



5.5.3.2 int DenseUnaryPixelPotential::Train (Dataset $*$ *dataset,* vector$<$ string $>$ & *imageList,* int *from,* int *to,* bool *FAST_COMPUTATION* = `false`) `[virtual]`

Train potential.

Parameters

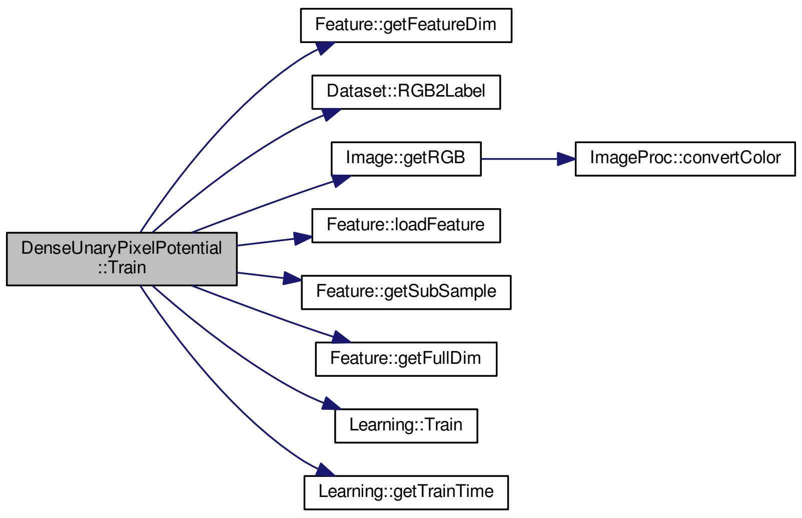
dataset	pointer to the current dataset
imageList	list of images
from	start index
to	stop index, set to -1 to process all images startin from "from"
FAST_COMPU-TATION	flag to activate fast computation

Returns

estimated training time in miliseconds

Implements Potential.

Here is the call graph for this function:



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

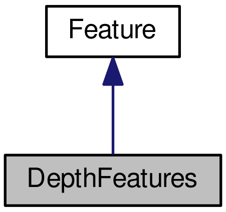
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Potential.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Potential.cpp

5.6 DepthFeatures Class Reference

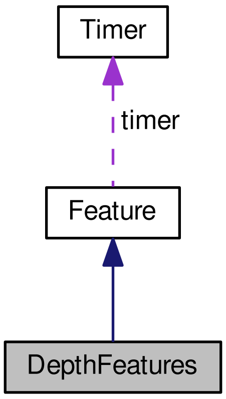
The DepthFeatures class.

```
#include <Feature.hpp>
```

Inheritance diagram for DepthFeatures:



Collaboration diagram for DepthFeatures:



Public Member Functions

- DepthFeatures (vector< Size > &all_patches, string depthFolder, string ext, int numSubSample)

  Class Constructor.
- void extractFeatures (Image ∗im, Mat &features)

  Extract features.

Additional Inherited Members

5.6.1 Detailed Description

The DepthFeatures class.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 DepthFeatures::DepthFeatures (vector< Size > & *all_patches*, string *depthFolder*, string *ext*, int *numSubSample*)

Class Constructor.

Parameters

all_patches	set of all patches
depthFolder	folder to store features
ext	extension of feature files
numSubSample	sub sample size

5.6.3 Member Function Documentation

5.6.3.1 void DepthFeatures::extractFeatures (Image ∗ *im*, Mat & *features*) `[virtual]`

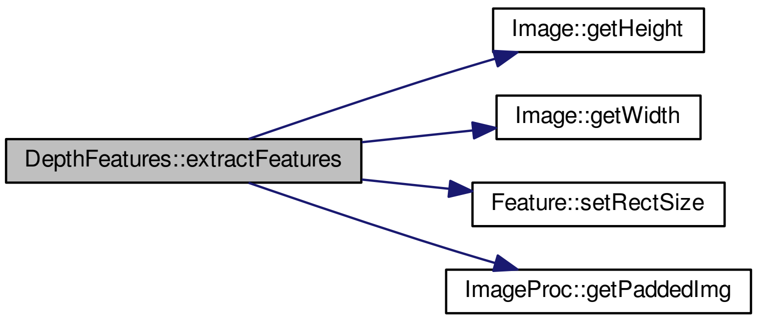
Extract features.

Parameters

im	image
features	feature matrix

Implements Feature.

Here is the call graph for this function:



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

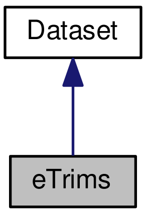
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Feature.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Feature.cpp

5.7 eTrims Class Reference

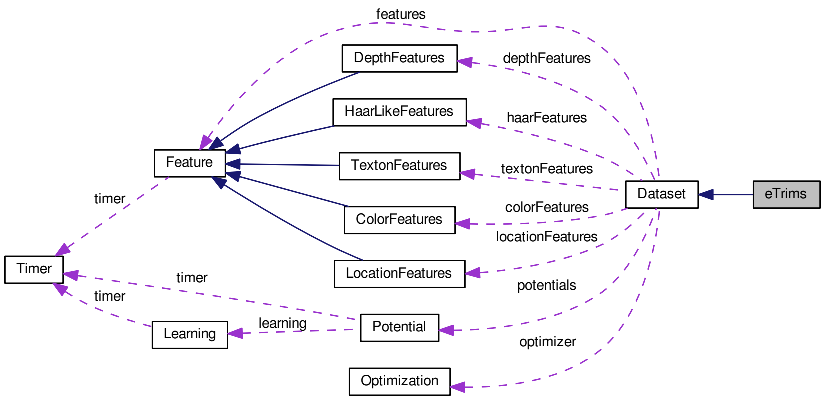
The eTrims class.

`#include <Dataset.hpp>`

Inheritance diagram for eTrims:



Collaboration diagram for eTrims:



Public Member Functions

- eTrims ()

  eTrims
- void RGB2Label (Mat rgb, Mat &labels)

  Convert from RGB to Label.
- void Label2RGB (Mat labels, Mat &rgb)

  Convert from label to RGB.

Additional Inherited Members

5.7.1 Detailed Description

The eTrims class.

5.7.2 Member Function Documentation

5.7.2.1 void eTrims::Label2RGB (Mat *labels*, Mat & *rgb*) [virtual]

Convert from label to RGB.

Parameters

labels	label matrix
rgb	RGB image

Reimplemented from Dataset.

5.7.2.2 void eTrims::RGB2Label (Mat *rgb*, Mat & *labels*) [virtual]

Convert from RGB to Label.

Parameters

rgb	RGB image
labels	label matrix to store labels for each pixel

Reimplemented from Dataset.

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

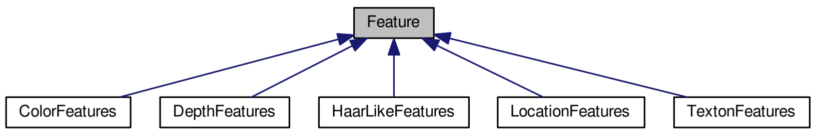
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Dataset.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Dataset.cpp

5.8 Feature Class Reference

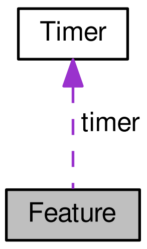
The Feature class.

`#include <Feature.hpp>`

Inheritance diagram for Feature:



Collaboration diagram for Feature:



Public Member Functions

- Feature ()

  Class Constructor.
- virtual ∼Feature ()

  Class Deconstructor.
- void saveFeature (string imName, Mat &features)

  Save features.
- void loadFeature (string imName, Mat &features)

  Load feature.
- virtual void extractFeatures (Image ∗im, Mat &features)=0

  Extract features.
- void setSubSample (int ss)

  Set sub sample size.
- int getSubSample ()

  Get sub sample size.
- int getFeatureDim ()

  Get size of feature set.
- int getFullDim ()

  Get size of full feature set.
- void setFeatureDim (int dim)

  Set size of feature set.

Public Attributes

- string name_

  name of the feature
- Mat isComputeFeature

  matrix to store feature indices indicate which features to be computed

Protected Member Functions

- void setRectSize (int height, int width)

  Set rectangle size.

Protected Attributes

- FILE ∗ foperator

  File operator.
- string ftFolder

  folder to store feature
- string ext_

  feature file extension
- int subSample

  size of sub sample
- int featureSize

  size of feature
- int featureDim

  size of selected feature set
- int rHeight

patch height
- int rWidth

  patch width
- int halfHeight

  half size of patch height
- int halfWidth

  half size of patch width
- int sixthOfHeight

  sixth of patch height
- int sixthOfWidth

  sixth of patch width
- int fourthOfHeight

  fourth of patch height
- int fourthOfWidth

  fourth of patch width
- float weight0

  weight of black region (Haar-like features)
- float weight1

  weight of white region (Haar-like features)
- vector$<$ Size $>$ allPatches

  vector of all patches
- Timer timer

  timer to estimate the time

5.8.1 Detailed Description

The Feature class.

5.8.2 Member Function Documentation

5.8.2.1 virtual void Feature::extractFeatures (Image $*$ *im*, Mat & *features*) `[pure virtual]`

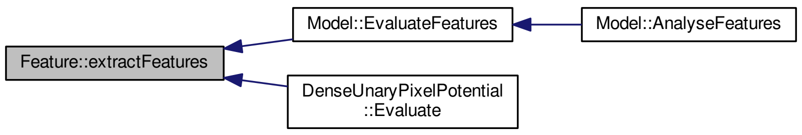
Extract features.

Parameters

im	image
features	feature matrix

Implemented in TextonFeatures, DepthFeatures, LocationFeatures, ColorFeatures, and HaarLikeFeatures.

Here is the caller graph for this function:



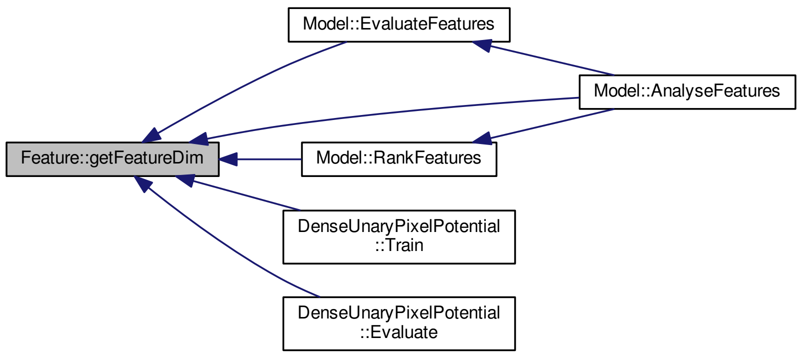
5.8.2.2 int Feature::getFeatureDim ( )

Get size of feature set.

Returns

size of feature set

Here is the caller graph for this function:



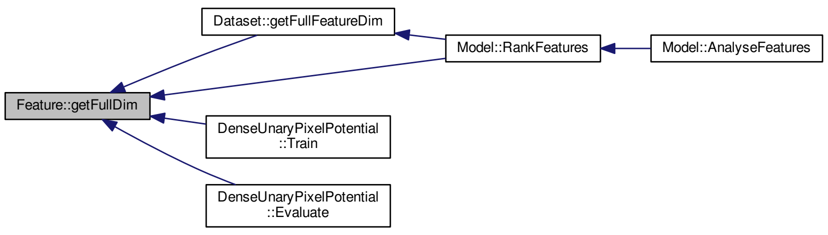
5.8.2.3 int Feature::getFullDim ( )

Get size of full feature set.

Returns

size of full feature set

Here is the caller graph for this function:


Dataset::getFullFeatureDim
Model::RankFeatures
Model::AnalyseFeatures
Feature::getFullDim
DenseUnaryPixelPotential ::Train
DenseUnaryPixelPotential ::Evaluate

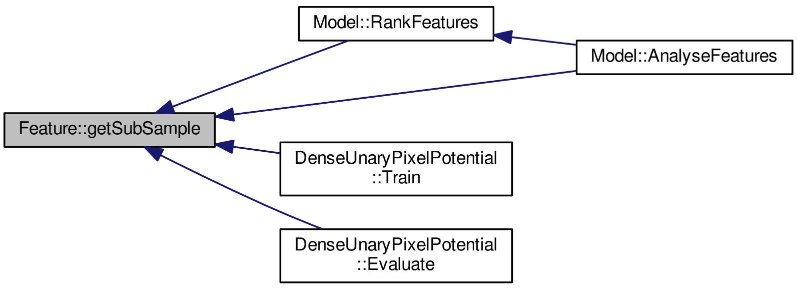
5.8.2.4 int Feature::getSubSample ( )

Get sub sample size.

Returns

size of sub sample

Here is the caller graph for this function:


Model::RankFeatures
Model::AnalyseFeatures
Feature::getSubSample
DenseUnaryPixelPotential ::Train
DenseUnaryPixelPotential ::Evaluate

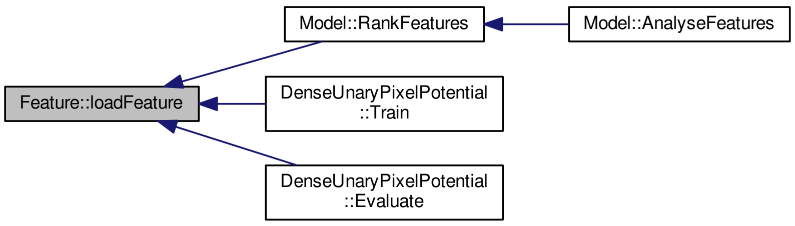
5.8.2.5 void Feature::loadFeature (string *imName*, Mat & *features*)

Load feature.

Parameters

imName	image name
features	feature matrix

Here is the caller graph for this function:


Model::RankFeatures
Model::AnalyseFeatures
Feature::loadFeature
DenseUnaryPixelPotential ::Train
DenseUnaryPixelPotential ::Evaluate

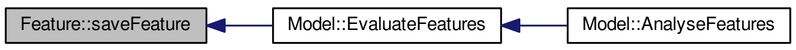
5.8.2.6 void Feature::saveFeature (string *imName*, Mat & *features*)

Save features.

Parameters

imName	image name
features	feature matrix

Here is the caller graph for this function:



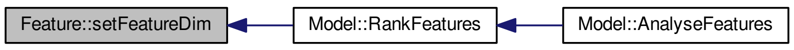
5.8.2.7 void Feature::setFeatureDim (int *dim*)

Set size of feature set.

Parameters

dim	size

Here is the caller graph for this function:



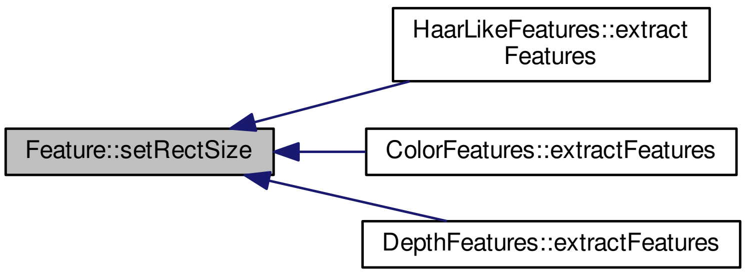
5.8.2.8 void Feature::setRectSize (int *height,* int *width*) [protected]

Set rectangle size.

Parameters

height	height of new rectangle
width	width of new rectangle

Here is the caller graph for this function:


HaarLikeFeatures::extract Features
Feature::setRectSize
ColorFeatures::extractFeatures
DepthFeatures::extractFeatures

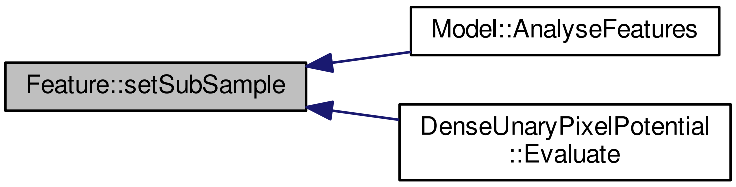
5.8.2.9 void Feature::setSubSample (int *ss*)

Set sub sample size.

Parameters

ss	sub sample size

Here is the caller graph for this function:



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

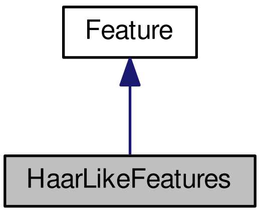
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Feature.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Feature.cpp

5.9 HaarLikeFeatures Class Reference

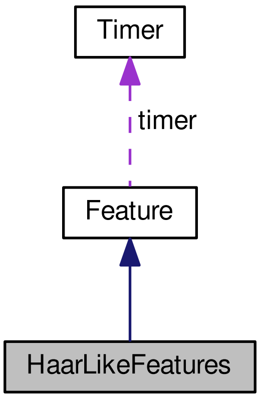
The HaarLikeFeatures class.

```
#include <Feature.hpp>
```

Inheritance diagram for HaarLikeFeatures:



Collaboration diagram for HaarLikeFeatures:



Public Member Functions

- HaarLikeFeatures (vector< Size > &all_patches, string haarFolder, string ext, int numSubSample)

  Class Constructor.
- void extractFeatures (Image ∗im, Mat &features)

  Extract features.

Protected Member Functions

- void horizontalEdge (Mat &im, int r, int c, float &fValue)

  Horizontal Edge Feature.
- void verticalEdge (Mat &im, int r, int c, float &fValue)

  Vertical Edge Feature.
- void horizontalLine (Mat &im, int r, int c, float &fValue)

  Horizontal Line Feature.
- void verticalLine (Mat &im, int r, int c, float &fValue)

  Vertical Edge Feature.
- void centerSurround (Mat &im, int r, int c, float &fValue)

  Center Surround Feature.
- void fourSquare (Mat &im, int r, int c, float &fValue)

  Four Square Feature.

Additional Inherited Members

5.9.1 Detailed Description

The HaarLikeFeatures class.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 HaarLikeFeatures::HaarLikeFeatures (vector$<$ Size $>$ & *all_patches,* string *haarFolder,* string *ext,* int *numSubSample*)

Class Constructor.

Parameters

all_patches	set of all patches
haarFolder	folder to store features
ext	extension of feature files
numSubSample	sub sample size

5.9.3 Member Function Documentation

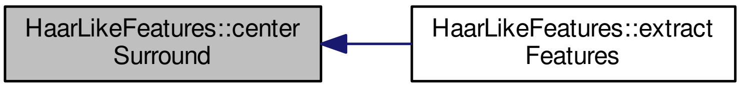
5.9.3.1 void HaarLikeFeatures::centerSurround (Mat & *im*, int *r*, int *c*, float & *fValue*) [protected]

Center Surround Feature.

Parameters

im	image
r	row of pixel at which the feature will be computed
c	column of pixel at which the feature will be computed
fValue	feature value

Here is the caller graph for this function:


HaarLikeFeatures::center Surround
HaarLikeFeatures::extract Features

5.9.3.2 void HaarLikeFeatures::extractFeatures (Image * *im*, Mat & *features*) [virtual]

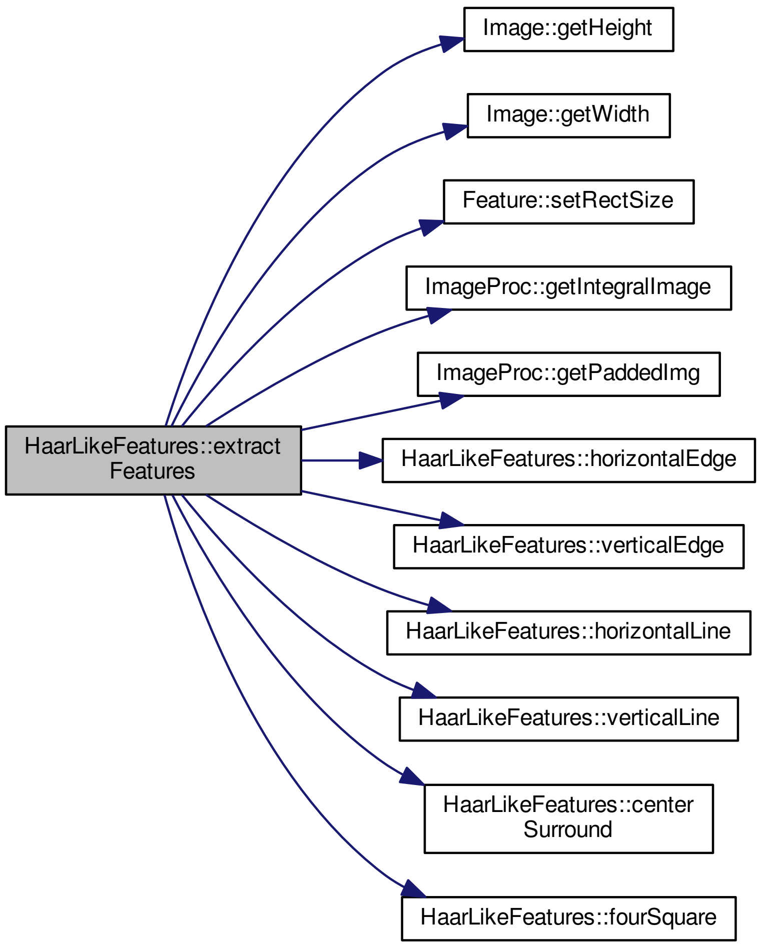
Extract features.

Parameters

im	image
features	feature matrix

Implements Feature.

Here is the call graph for this function:


HaarLikeFeatures::extract Features
Image::getHeight
Image::getWidth
Feature::setRectSize
ImageProc::getIntegralImage
ImageProc::getPaddedImg
HaarLikeFeatures::horizontalEdge
HaarLikeFeatures::verticalEdge
HaarLikeFeatures::horizontalLine
HaarLikeFeatures::verticalLine
HaarLikeFeatures::center Surround
HaarLikeFeatures::fourSquare

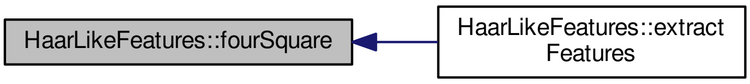
5.9.3.3 void HaarLikeFeatures::fourSquare (Mat & *im,* int *r,* int *c,* float & *fValue*) `[protected]`

Four Square Feature.

Parameters

im	image
r	row of pixel at which the feature will be computed
c	column of pixel at which the feature will be computed
fValue	feature value

Here is the caller graph for this function:


HaarLikeFeatures::fourSquare
HaarLikeFeatures::extract Features

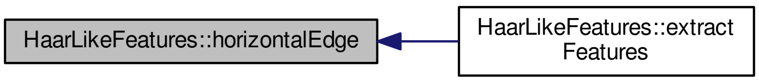
5.9.3.4 void HaarLikeFeatures::horizontalEdge (Mat & *im*, int *r*, int *c*, float & *fValue*) [protected]

Horizontal Edge Feature.

Parameters

im	image
r	row of pixel at which the feature will be computed
c	column of pixel at which the feature will be computed
fValue	feature value

Here is the caller graph for this function:


HaarLikeFeatures::horizontalEdge
HaarLikeFeatures::extract Features

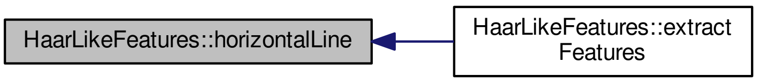
5.9.3.5 void HaarLikeFeatures::horizontalLine (Mat & *im*, int *r*, int *c*, float & *fValue*) [protected]

Horizontal Line Feature.

Parameters

im	image
r	row of pixel at which the feature will be computed
c	column of pixel at which the feature will be computed
fValue	feature value

Here is the caller graph for this function:


HaarLikeFeatures::horizontalLine
HaarLikeFeatures::extract Features

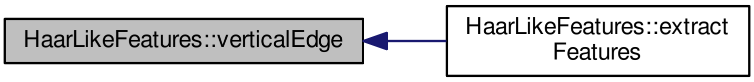
5.9.3.6 void HaarLikeFeatures::verticalEdge (Mat & *im*, int *r*, int *c*, float & *fValue*) `[protected]`

Vertical Edge Feature.

Parameters

im	image
r	row of pixel at which the feature will be computed
c	column of pixel at which the feature will be computed
fValue	feature value

Here is the caller graph for this function:


HaarLikeFeatures::verticalEdge
HaarLikeFeatures::extract Features

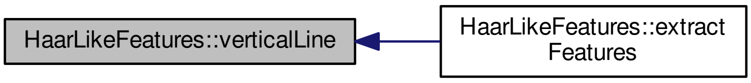
5.9.3.7 void HaarLikeFeatures::verticalLine (Mat & *im*, int *r*, int *c*, float & *fValue*) `[protected]`

Vertical Edge Feature.

Parameters

im	image
r	row of pixel at which the feature will be computed
c	column of pixel at which the feature will be computed
fValue	feature value

Here is the caller graph for this function:


HaarLikeFeatures::verticalLine
HaarLikeFeatures::extract Features

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

- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Feature.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Feature.cpp

5.10 Image Class Reference

The Image class.

`#include <Image.hpp>`

Public Member Functions

- Image (string impath, string depthpath="")
  Class Constructor.
- ∼Image ()
  Class DeConstructor.
- int getWidth ()
  get image width
- int getHeight ()
  get image height
- Mat getImage ()
  get image
- Mat getRGB ()
  get RGB image
- bool isEmpty ()
  is image empty ?

Public Attributes

- string imName
  image path
- Mat BGRImage_
  BGR image.
- Mat LABImage_
  Lab image.
- Mat L
  L channel.
- Mat L8U
  L channel, unsigned 8 bit.
- Mat a
  a channel
- Mat b
  b channel
- Mat depth
  depth image

5.10.1 Detailed Description

The Image class.

5.10.2 Constructor & Destructor Documentation

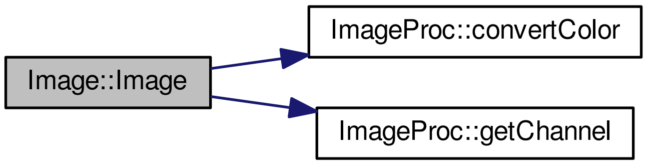
5.10.2.1 Image::Image (string *impath*, string *depthpath* = "")

Class Constructor.

Parameters

impath	Path to the image
depthpath	Path to the depth image

Here is the call graph for this function:



5.10.3 Member Function Documentation

5.10.3.1 int Image::getHeight ( )

get image height

Returns

image height

Here is the caller graph for this function:

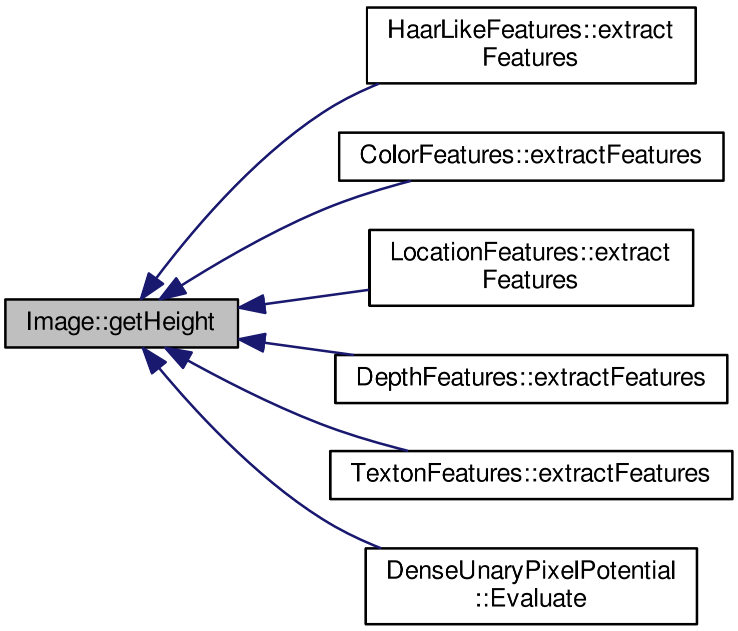


Image::getHeight
HaarLikeFeatures::extract Features
ColorFeatures::extractFeatures
LocationFeatures::extract Features
DepthFeatures::extractFeatures
TextonFeatures::extractFeatures
DenseUnaryPixelPotential ::Evaluate

5.10.3.2 Mat Image::getImage ( )

get image

Returns

image

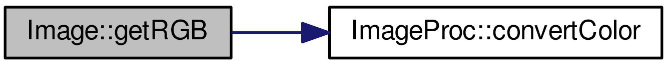
5.10.3.3 Mat Image::getRGB ( )

get RGB image

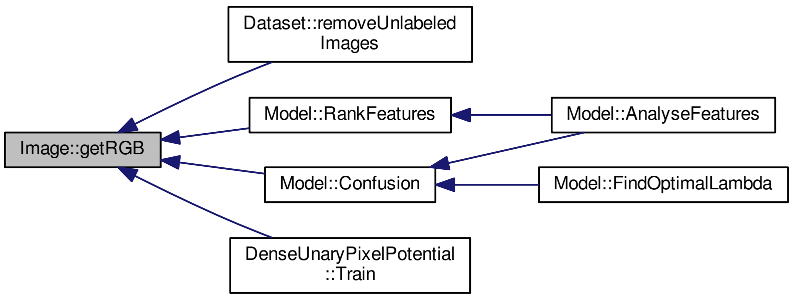
Returns

RGB image

Here is the call graph for this function:



Here is the caller graph for this function:



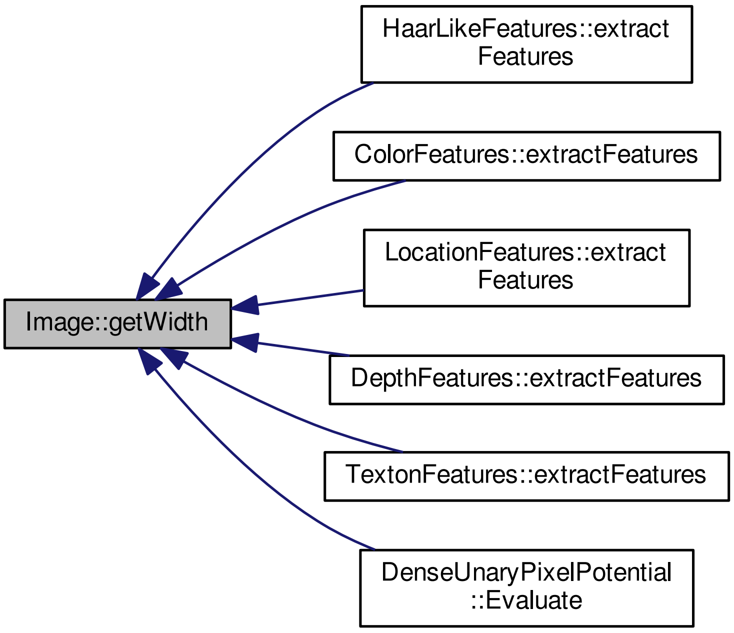
5.10.3.4 int Image::getWidth ( )

get image width

Returns

image width

Here is the caller graph for this function:


HaarLikeFeatures::extract Features
ColorFeatures::extractFeatures
LocationFeatures::extract Features
Image::getWidth
DepthFeatures::extractFeatures
TextonFeatures::extractFeatures
DenseUnaryPixelPotential ::Evaluate

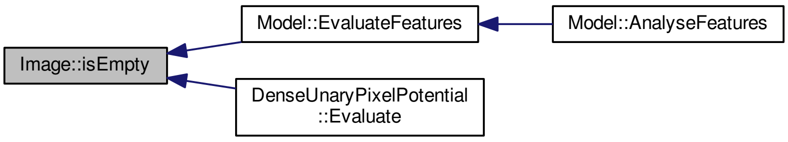
5.10.3.5 bool Image::isEmpty ( )

is image empty ?

Returns

true or false

Here is the caller graph for this function:



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

- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Image.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Image.cpp

5.11 ImageProc Class Reference

The Image Processing class.

`#include <ImageProc.hpp>`

Public Member Functions

- ImageProc ()

  Class Constructor.

Static Public Member Functions

- static Mat getEdgeImage (Mat input)

  compute image edges
- static Mat getIntegralImage (Mat input)

  compute integral image
- static Mat getPaddedImg (Mat input, int extTopRows, int extBotRows, int extLeftCols, int extRightCols)

  zero pad to the image
- static Mat convertColor (Mat input, int code)

  convert image color space
- static Mat getChannel (Mat input, unsigned char channel)

  get specific channel of image

5.11.1 Detailed Description

The Image Processing class.

5.11.2 Member Function Documentation

5.11.2.1 Mat ImageProc::convertColor (Mat *input,* int *code*) `[static]`

convert image color space

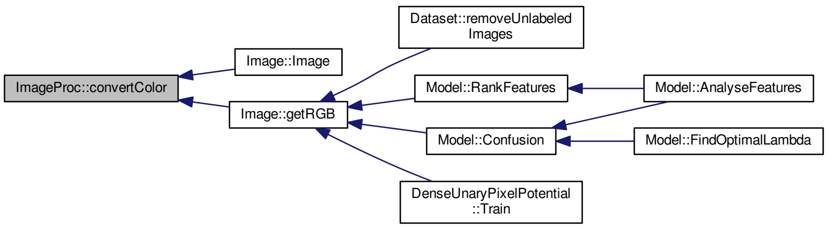
Parameters

input	image
code	OpenCV color conversion code

Returns

color converted image

Here is the caller graph for this function:



5.11.2.2 Mat ImageProc::getChannel (Mat *input*, unsigned char *channel* = `0`) `[static]`

get specific channel of image

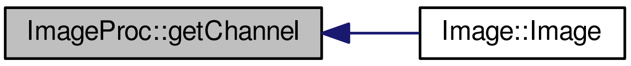
Parameters

input	image
channel	channel to return

Returns

image channel

Here is the caller graph for this function:



5.11.2.3 Mat ImageProc::getEdgeImage (Mat *input*) `[static]`

compute image edges

Parameters

input	RGB image

Returns

edge image

5.11.2.4 Mat ImageProc::getIntegralImage (Mat *input*) [static]

compute integral image

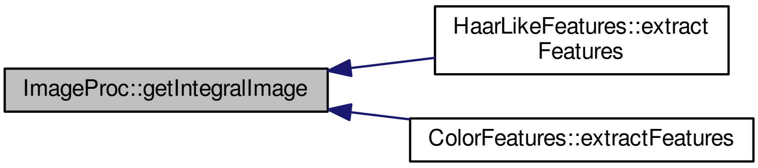
Parameters

input	RGB Image

Returns

integral image

Here is the caller graph for this function:


HaarLikeFeatures::extract Features
ImageProc::getIntegralImage
ColorFeatures::extractFeatures

5.11.2.5 Mat ImageProc::getPaddedImg (Mat *input,* int *extTopRows,* int *extBotRows,* int *extLeftCols,* int *extRightCols*) [static]

zero pad to the image

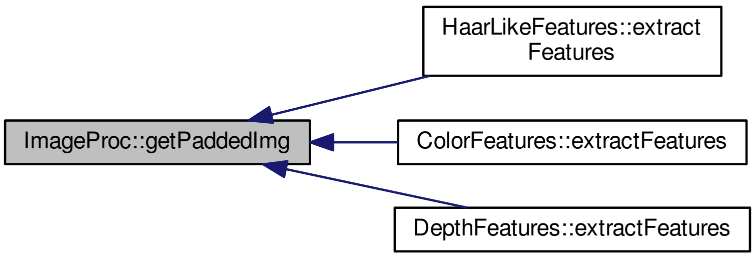
Parameters

input	image
extTopRows	number of top rows to pad zero
extBotRows	number of bot rows to pad zero
extLeftCols	number of left cols to pad zero
extRightCols	number of right cols to pad zero

Returns

zero padded image

Here is the caller graph for this function:


HaarLikeFeatures::extract Features
ImageProc::getPaddedImg
ColorFeatures::extractFeatures
DepthFeatures::extractFeatures

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

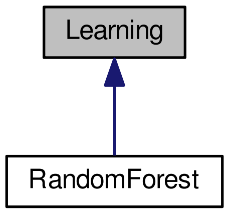
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/ImageProc.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/ImageProc.cpp

5.12 Learning Class Reference

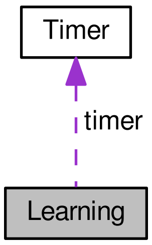
Virtual Class for Learning Algorithms.

`#include <Learning.hpp>`

Inheritance diagram for Learning:



Collaboration diagram for Learning:



Public Member Functions

- Learning ()

  Class Constructor.
- virtual ∼Learning ()

  Class Deconstructor.
- virtual void Train (Mat &trainData, const Mat &labels=Mat())=0

  Train Classifier.
- virtual void Evaluate (Mat &testData, Mat &possibleLabels, Mat &labelProbs)=0

  Predict class probabilities.
- virtual void SaveClassifier ()=0

Save Classifier.
- virtual void LoadClassifier ()=0

  Load Classifier.
- virtual void ClearClassifier ()=0

  Clear classifier object.
- int getTrainTime ()

  Return estimated training time.

Public Attributes

- string trainFile

  Path to the classifier file.
- string name_

  Name of the classifier.

Protected Attributes

- uint nClass

  Total number of classes.
- Timer timer

  Timer to estimate the time.
- int train_time

  Training time in miliseconds.

5.12.1 Detailed Description

Virtual Class for Learning Algorithms.

5.12.2 Member Function Documentation

5.12.2.1 virtual void Learning::Evaluate (Mat & *testData*, Mat & *possibleLabels*, Mat & *labelProbs*) [pure virtual]

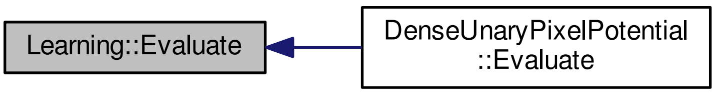
Predict class probabilities.

Parameters

testData	Test data, size of {number of samples x number of features}
possibleLabels	argmax< class probabilities> for each test sample
labelProbs	Class probabilities for each test sample

Implemented in RandomForest.

Here is the caller graph for this function:



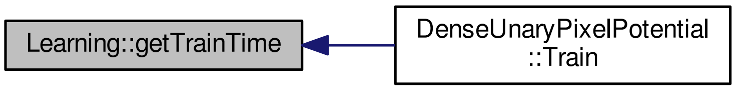
5.12.2.2 int Learning::getTrainTime ( )

Return estimated training time.

Returns

Estimated time in miliseconds

Here is the caller graph for this function:



5.12.2.3 virtual void Learning::Train (Mat & *trainData*, const Mat & *labels* = `Mat ()`) `[pure virtual]`

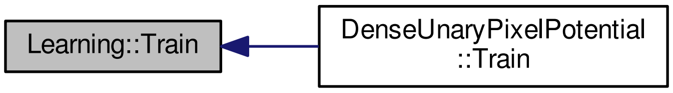
Train Classifier.

Parameters

trainData	Training data, size of {number of samples x number of features}
labels	Ground truth class labels, size of {1 x number of samples}

Implemented in RandomForest.

Here is the caller graph for this function:



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

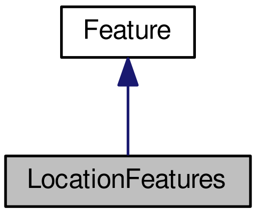
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Learning.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Learning.cpp

5.13 LocationFeatures Class Reference

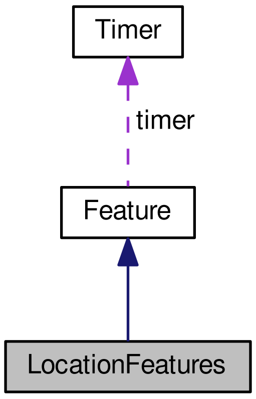
The LocationFeatures class.

```
#include <Feature.hpp>
```

Inheritance diagram for LocationFeatures:



Collaboration diagram for LocationFeatures:



Public Member Functions

- LocationFeatures (string LocFolder, string ext, int numSubSample)

    Class Constructor.
- void extractFeatures (Image ∗im, Mat &features)

    Extract features.

Additional Inherited Members

5.13.1 Detailed Description

The LocationFeatures class.

5.13.2 Constructor & Destructor Documentation

5.13.2.1 LocationFeatures::LocationFeatures (string *LocFolder,* string *ext,* int *numSubSample*)

Class Constructor.

Parameters

LocFolder	folder to store features
ext	extension of feature files
numSubSample	sub sample size

5.13.3 Member Function Documentation

5.13.3.1 void LocationFeatures::extractFeatures (Image ∗ *im*, Mat & *features*) [virtual]

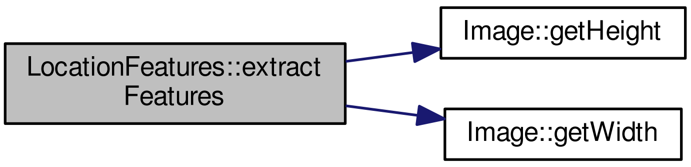
Extract features.

Parameters

im	image
features	feature matrix

Implements Feature.

Here is the call graph for this function:



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

- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Feature.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Feature.cpp

5.14 Model Class Reference

The Model class.

`#include <Model.hpp>`

Public Member Functions

- Model ()

  Class Constructor.
- void SetStructure (Dataset ∗dataset)

  Set model structure.
- void EvaluateFeatures (vector< string > &imageList, int from=0, int to=-1, int &eval_ftr_time=DummyTime)

  Evaluate features.
- void TrainPotentials (vector< string > &imageList, int from=0, int to=-1, int &train_pt_time=DummyTime)

  Train potentials.

- void EvaluatePotentials (vector< string > &imageList, int from=0, int to=-1, int &eval_pt_time=DummyTime)

  Evaluate potentials.
- void Confusion (vector< string > &imageList, string folder, string confFileName, int from=0, int to=-1)

  Compute confusion matrix.
- void Solve (vector< string > &imageList, int from=0, int to=-1, int &solving_time=DummyTime)

  Solve optimization.
- void RankFeatures (vector< string > &imageList, int from=0, int to=-1, bool isRank=true, int &ranking_-time=DummyTime)

  Rank features with mrmr.
- void AnalyseFeatures ()

  Analyse features.
- int SelectFeatures (float alpha, float beta)

  Select features.
- void FindOptimalLambda (vector< string > &imageList, int from=0, int to=-1)

  Find optimal lambda for VariationalOptimization.
- void SavePotentialMap (vector< string > &imageList, int from=0, int to=-1)

  Save potential map for images.
- void ActivateFastComputation (bool isFast=true)

  Activate fast computation Load features instead of computing on the fly.
- void PrintDatasetInfo ()

  Print dataset info.
- Performance getPERFORMANCE ()

  get current performance

5.14.1 Detailed Description

The Model class.

5.14.2 Member Function Documentation

5.14.2.1 void Model::ActivateFastComputation (bool *isFast* = `true`)

Activate fast computation Load features instead of computing on the fly.

Activate the loading only relevant features from pre-computed feature files.

Parameters

isFast	flag to activate fast computation

5.14.2.2 void Model::Confusion (vector< string > & *imageList*, string *folder*, string *confFileName*, int *from* = `0`, int *to* = `-1`)

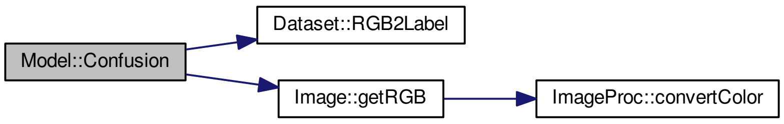
Compute confusion matrix.

Parameters

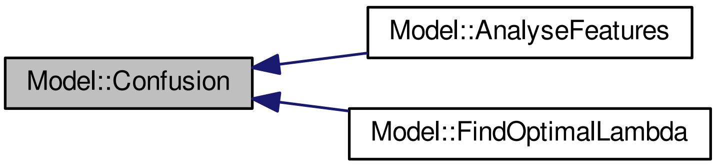
imageList	list of images
folder	path to the folder to save confusion matrix
confFileName	file name for confusion matrix

from	start index
to	stop index, set to -1 to segment all images starting from "from"

Here is the call graph for this function:



Here is the caller graph for this function:



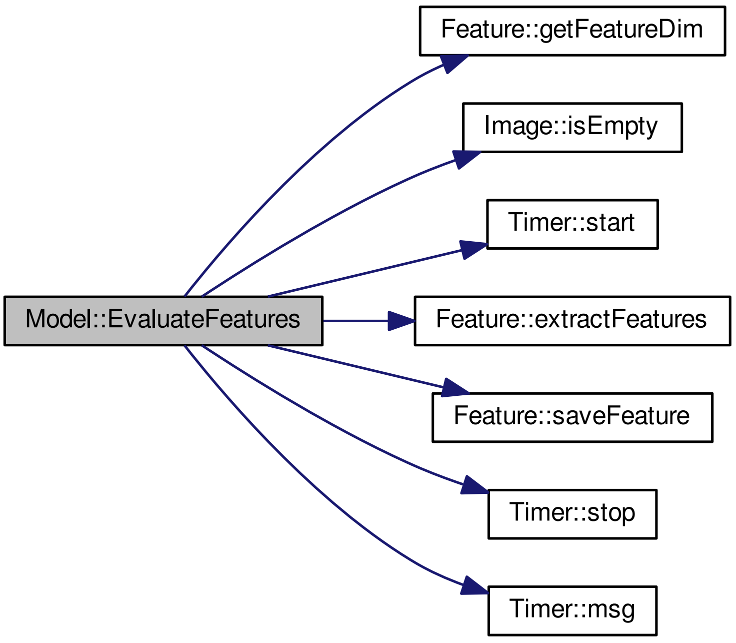
5.14.2.3 void Model::EvaluateFeatures (vector< string > & *imageList*, int *from* = `0`, int *to* = `-1`, int & *eval_ftr_time* = `DummyTime`)

Evaluate features.

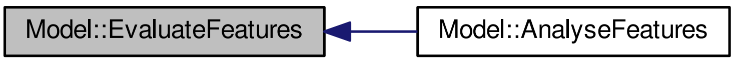
Parameters

imageList	list of images
from	start index
to	stop index, set to -1 to segment all images starting from "from"
eval_ftr_time	estimated time per image in miliseconds

Here is the call graph for this function:



Here is the caller graph for this function:



5.14.2.4 void Model::EvaluatePotentials (vector< string > & *imageList*, int *from* = `0`, int *to* = `-1`, int & *eval_pt_time* = `DummyTime`)

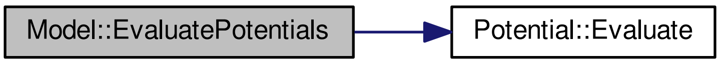
Evaluate potentials.

Parameters

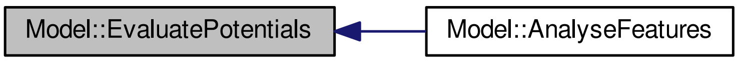
imageList	list of images
from	start index
to	stop index, set to -1 to segment all images starting from "from"

eval_pt_time	estimated time per image in miliseconds

Here is the call graph for this function:



Here is the caller graph for this function:



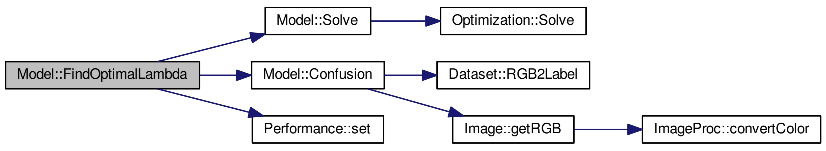
5.14.2.5 void Model::FindOptimalLambda (vector< string > & *imageList*, int *from* = 0, int *to* = -1)

Find optimal lambda for VariationalOptimization.

Parameters

imageList	list of images
from	start index
to	stop index, set to -1 to segment all images starting from "from"

Here is the call graph for this function:



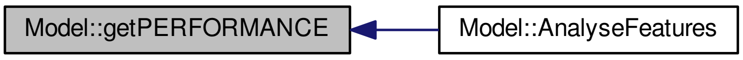
5.14.2.6 Performance Model::getPERFORMANCE ( )

get current performance

Returns

current performance

Here is the caller graph for this function:



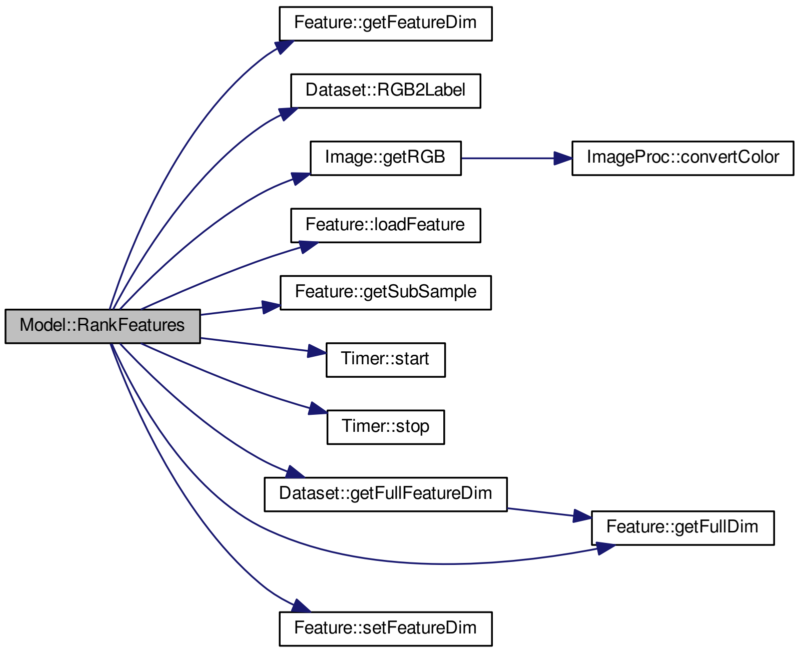
5.14.2.7 void Model::RankFeatures (vector< string > & *imageList*, int *from* = `0`, int *to* = `-1`, bool *isRank* = `true`, int & *ranking_time* = `DummyTime`)

Rank features with mrmr.

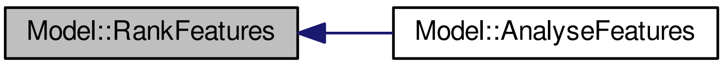
Parameters

imageList	list of images
from	start index
to	stop index, set to -1 to segment all images starting from "from"
isRank	flag to rank features
ranking_time	estimated time to rank features

Here is the call graph for this function:



Here is the caller graph for this function:



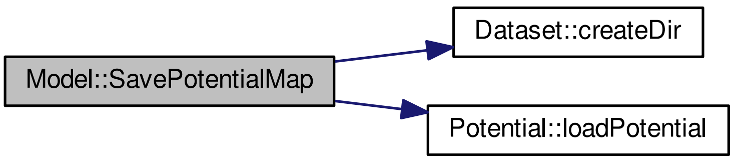
5.14.2.8 void Model::SavePotentialMap (vector< string > & *imageList*, int *from* = 0, int *to* = -1)

Save potential map for images.

Parameters

imageList	list of images
from	start index
to	stop index, set to -1 to segment all images starting from "from"

Here is the call graph for this function:



5.14.2.9 int Model::SelectFeatures (float *alpha*, float *beta*)

Select features.

Parameters

alpha	alpha
beta	beta

Returns

number of selected features

5.14.2.10 void Model::SetStructure (Dataset * *dataset*)

Set model structure.

Parameters

dataset	pointer to the dataset

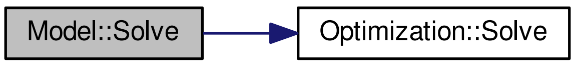
5.14.2.11 void Model::Solve (vector< string > & *imageList*, int *from* = `0`, int *to* = `-1`, int & *solving_time* = `DummyTime`)

Solve optimization.

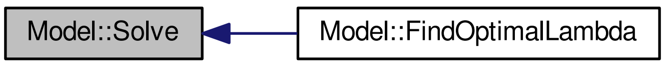
Parameters

imageList	list of images
from	start index
to	stop index, set to -1 to segment all images starting from "from"
solving_time	estimated time per image in miliseconds

Here is the call graph for this function:



Here is the caller graph for this function:



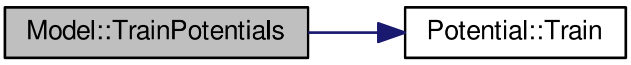
5.14.2.12 void Model::TrainPotentials (vector< string > & *imageList*, int *from* = `0`, int *to* = `-1`, int & *train_pt_time* = `DummyTime`)

Train potentials.

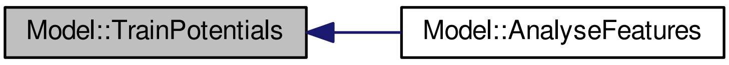
Parameters

imageList	list of images
from	start index
to	stop index, set to -1 to segment all images starting from "from"
train_pt_time	estimated time in miliseconds

Here is the call graph for this function:



Here is the caller graph for this function:



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

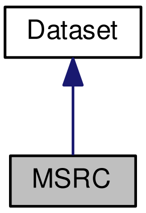
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Model.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Model.cpp

5.15 MSRC Class Reference

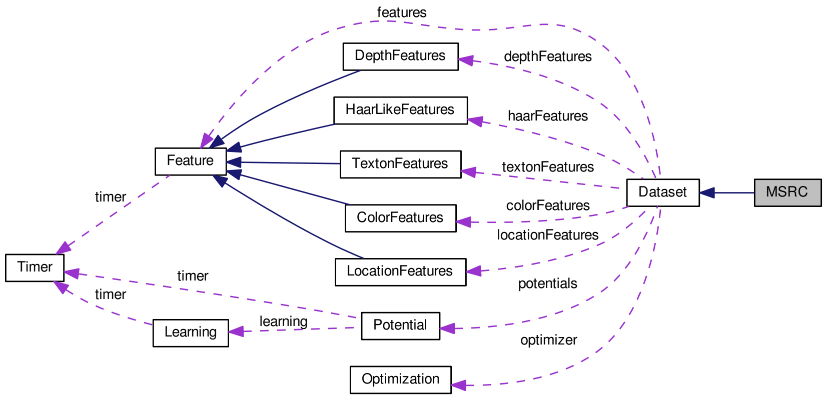
The MSRC class.

`#include <Dataset.hpp>`

Inheritance diagram for MSRC:



Collaboration diagram for MSRC:



Public Member Functions

- MSRC ()

  MSRC.
- void RGB2Label (Mat rgb, Mat &labels)

  Convert from RGB to Label.
- void Label2RGB (Mat labels, Mat &rgb)

  Convert from label to RGB.

Additional Inherited Members

5.15.1 Detailed Description

The MSRC class.

5.15.2 Member Function Documentation

5.15.2.1 void MSRC::Label2RGB (Mat *labels*, Mat & *rgb*) [virtual]

Convert from label to RGB.

Parameters

labels	label matrix
rgb	RGB image

Reimplemented from Dataset.

5.15.2.2 void MSRC::RGB2Label (Mat *rgb*, Mat & *labels*) [virtual]

Convert from RGB to Label.

Parameters

rgb	RGB image
labels	label matrix to store labels for each pixel

Reimplemented from Dataset.

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

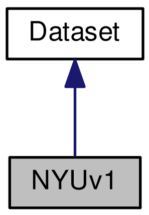
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Dataset.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Dataset.cpp

5.16 NYUv1 Class Reference

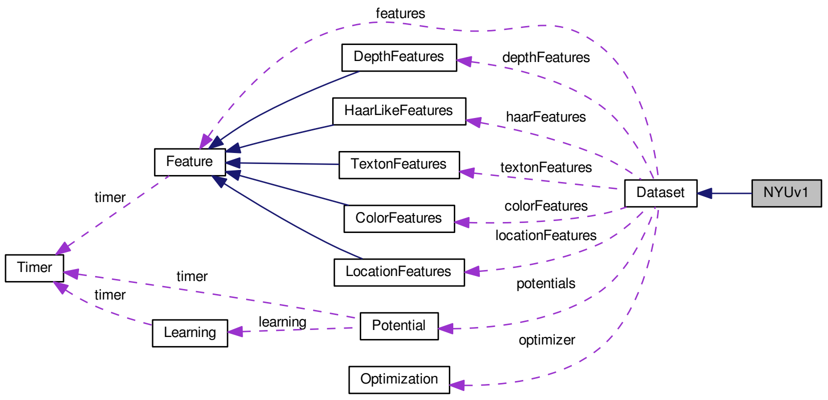
The NYUv1 class.

```
#include <Dataset.hpp>
```

Inheritance diagram for NYUv1:



Collaboration diagram for NYUv1:



Public Member Functions

- NYUv1 ()

Class Constructor.

- void RGB2Label (Mat rgb, Mat &labels)

  Convert from RGB to Label.

- void Label2RGB (Mat labels, Mat &rgb)

  Convert from label to RGB.

Additional Inherited Members

5.16.1 Detailed Description

The NYUv1 class.

5.16.2 Member Function Documentation

5.16.2.1 void NYUv1::Label2RGB (Mat *labels*, Mat & *rgb*) `[virtual]`

Convert from label to RGB.

Parameters

labels	label matrix
rgb	RGB image

Reimplemented from Dataset.

5.16.2.2 void NYUv1::RGB2Label (Mat *rgb*, Mat & *labels*) `[virtual]`

Convert from RGB to Label.

Parameters

rgb	RGB image
labels	label matrix to store labels for each pixel

Reimplemented from Dataset.

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

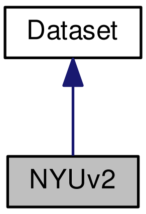
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Dataset.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Dataset.cpp

5.17 NYUv2 Class Reference

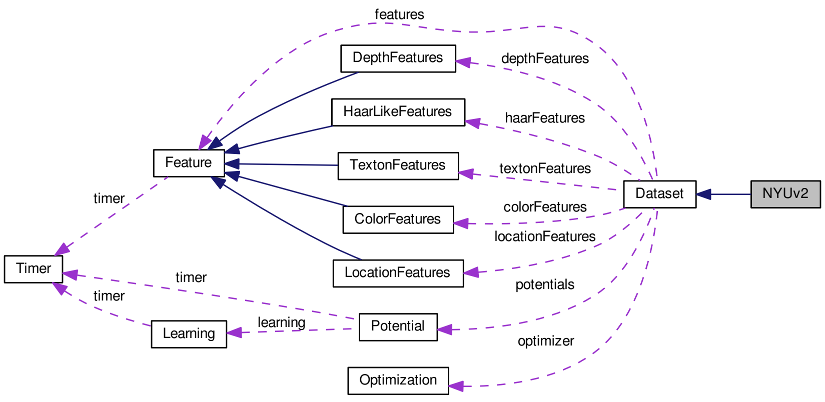
The NYUv2 class.

`#include <Dataset.hpp>`

Inheritance diagram for NYUv2:



Collaboration diagram for NYUv2:



Public Member Functions

- NYUv2 ()

  Class Constructor.
- void RGB2Label (Mat rgb, Mat &labels)

  Convert from RGB to Label.
- void Label2RGB (Mat labels, Mat &rgb)

  Convert from label to RGB.

Additional Inherited Members

5.17.1 Detailed Description

The NYUv2 class.

5.17.2 Member Function Documentation

5.17.2.1 void NYUv2::Label2RGB (Mat *labels*, Mat & *rgb*) `[virtual]`

Convert from label to RGB.

Parameters

labels	label matrix
rgb	RGB image

Reimplemented from Dataset.

5.17.2.2 void NYUv2::RGB2Label (Mat *rgb,* Mat & *labels*) [virtual]

Convert from RGB to Label.

Parameters

rgb	RGB image
labels	label matrix to store labels for each pixel

Reimplemented from Dataset.

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

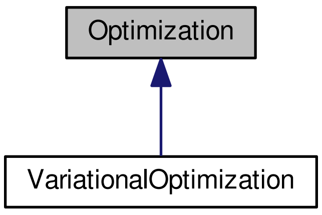
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Dataset.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Dataset.cpp

5.18 Optimization Class Reference

The Optimization class.

`#include <Optimization.hpp>`

Inheritance diagram for Optimization:



Public Member Functions

- virtual int Solve (Dataset $*$dataset, vector< string > &imageList, int from, int to, bool printTime=true)=0

  optimize solution

5.18.1 Detailed Description

The Optimization class.

5.18.2 Member Function Documentation

5.18.2.1 virtual int Optimization::Solve (Dataset ∗ *dataset*, vector< string > & *imageList*, int *from*, int *to*, bool *printTime* = `true`) `[pure virtual]`

optimize solution

Parameters

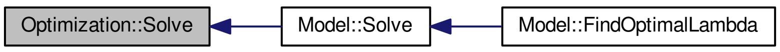
dataset	pointer to the current dataset
imageList	list of images to segment
from	start index
to	stop index, set to -1 to segment all images starting from "from"
printTime	flag to print out the estiamted time on output

Returns

estimated time per image in miliseconds

Implemented in VariationalOptimization.

Here is the caller graph for this function:



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

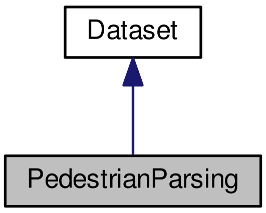
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Optimization.hpp

5.19 PedestrianParsing Class Reference

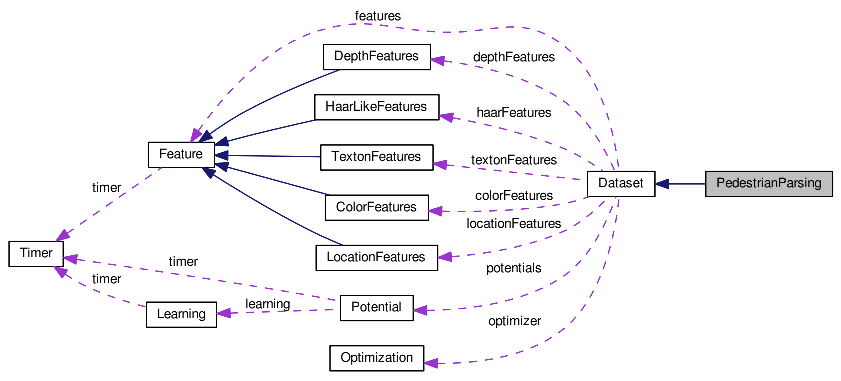
The PedestrianParsing class.

```
#include <Dataset.hpp>
```

Inheritance diagram for PedestrianParsing:



Collaboration diagram for PedestrianParsing:



Public Member Functions

- PedestrianParsing ()

  Class Constructor.
- void RGB2Label (Mat rgb, Mat &labels)

  Convert from RGB to Label.
- void Label2RGB (Mat labels, Mat &rgb)

  Convert from label to RGB.

Additional Inherited Members

5.19.1 Detailed Description

The PedestrianParsing class.

5.19.2 Member Function Documentation

5.19.2.1 void PedestrianParsing::Label2RGB (Mat *labels*, Mat & *rgb*) [virtual]

Convert from label to RGB.

Parameters

labels	label matrix
rgb	RGB image

Reimplemented from Dataset.

5.19.2.2 void PedestrianParsing::RGB2Label (Mat *rgb*, Mat & *labels*) [virtual]

Convert from RGB to Label.

Parameters

rgb	RGB image
labels	label matrix to store labels for each pixel

Reimplemented from Dataset.

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

- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Dataset.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Dataset.cpp

5.20 Performance Class Reference

The Performance class.

`#include <common.hpp>`

Public Member Functions

- Performance (double init=0)

  Class Constructor.
- void set (Performance perf)

  set performance
- Performance & operator= (const Performance &p)

  operator =
- Performance & operator+= (const Performance &p)

  operator +=
- Performance & operator/= (const int &t)

  operator /=

Public Attributes

- double overall

  overall score
- double average

  average score
- double waverage

  weighted average score

5.20.1 Detailed Description

The Performance class.

5.20.2 Constructor & Destructor Documentation

5.20.2.1 Performance::Performance (double *init* = `0`) `[inline]`

Class Constructor.

Parameters

init	initial performance score

5.20.3 Member Function Documentation

5.20.3.1 Performance& Performance::operator+= (const Performance & *p*) [inline]

operator +=

Parameters

p	pointer to performance

Returns

new performance

5.20.3.2 Performance& Performance::operator/= (const int & *t*) [inline]

operator /=

Parameters

t	number

Returns

new performance

5.20.3.3 Performance& Performance::operator= (const Performance & *p*) [inline]

operator =

Parameters

p	pointer to performance

Returns

new performance

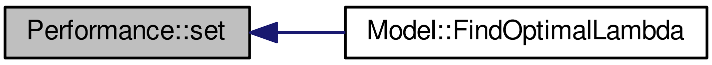
5.20.3.4 void Performance::set (Performance *perf*) [inline]

set performance

Parameters

perf	performance

Here is the caller graph for this function:



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

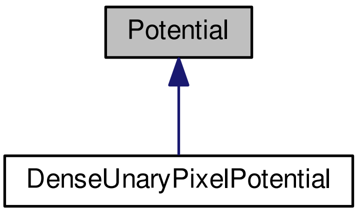
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/common.hpp

5.21 Potential Class Reference

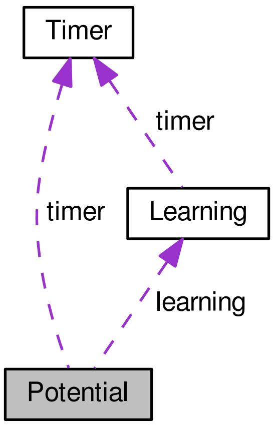
The Potential class.

```
#include <Potential.hpp>
```

Inheritance diagram for Potential:



Collaboration diagram for Potential:



Public Member Functions

- Potential ()

  Class Constructor.
- virtual ∼Potential ()

  Class Deconstructor.
- virtual int Train (Dataset ∗dataset, vector< string > &imageList, int from, int to, bool FAST_COMPUTATIO-N)=0

  Train potential.
- virtual int Evaluate (Dataset ∗dataset, vector< string > &imageList, int from, int to, bool FAST_COMPUTAT-ION)=0

  Evaluate potentials.
- void savePotential (string imName, Mat &potentials, int classNo)

  Save potential.
- void loadPotential (string imName, Mat &potentials)

  Load Potential.

Public Attributes

- string name_

  Potential name.
- Learning ∗ learning

  pointer to the learning algorithm

Protected Attributes

- FILE ∗ foperator

  file operator
- string folder

potential output folder

- string ext

    potential extension

- Timer timer

    timer to estimate the time

5.21.1 Detailed Description

The Potential class.

5.21.2 Member Function Documentation

5.21.2.1 virtual int Potential::Evaluate (Dataset $*$ *dataset,* vector$<$ string $>$ & *imageList,* int *from,* int *to,* bool *FAST_COMPUTATION*) `[pure virtual]`

Evaluate potentials.

Parameters

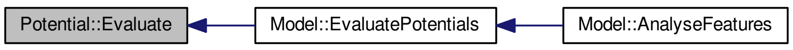
dataset	pointer to the current dataset
imageList	list of images
from	start index
to	stop index, set to -1 to process all images startin from "from"
FAST_COMPU-TATION	flag to activate fast computation

Returns

estimated time per image in miliseconds

Implemented in DenseUnaryPixelPotential.

Here is the caller graph for this function:



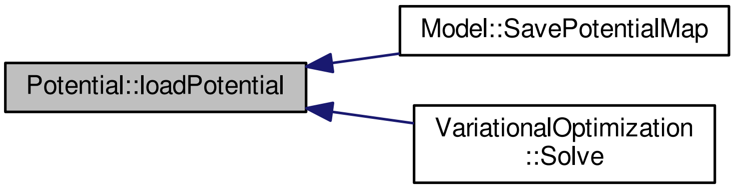
5.21.2.2 void Potential::loadPotential (string *imName,* Mat & *potentials*)

Load Potential.

Parameters

imName	path to the image
potentials	potentials

Here is the caller graph for this function:



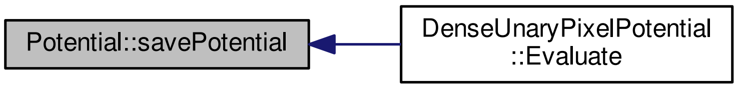
5.21.2.3 void Potential::savePotential (string *imName,* Mat & *potentials,* int *classNo*)

Save potential.

Parameters

imName	path to the image
potentials	potentials
classNo	total number of classes

Here is the caller graph for this function:



5.21.2.4 virtual int Potential::Train (Dataset ∗ *dataset,* vector< string > & *imageList,* int *from,* int *to,* bool *FAST_COMPUTATION*) `[pure virtual]`

Train potential.

Parameters

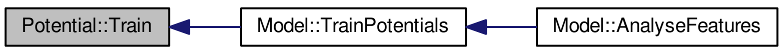
dataset	pointer to the current dataset
imageList	list of images
from	start index
to	stop index, set to -1 to process all images startin from "from"
FAST_COMPUTATION	flag to activate fast computation

Returns

estimated training time in miliseconds

Implemented in DenseUnaryPixelPotential.

Here is the caller graph for this function:



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

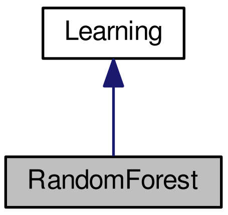
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Potential.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Potential.cpp

5.22 RandomForest Class Reference

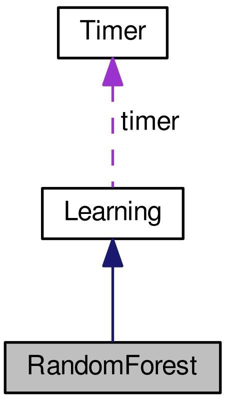
The RandomForest class.

`#include <Learning.hpp>`

Inheritance diagram for RandomForest:



Collaboration diagram for RandomForest:



Public Member Functions

- RandomForest (string clsFile, CvRTParams params, unsigned int numClasses)

  Class Constructor.
- void Train (Mat &trainData, const Mat &labels=Mat())

  Train Classifier.
- void Evaluate (Mat &testData, Mat &possibleLabels, Mat &labelProbs)

  Predict class probabilities.
- void SaveClassifier ()

  Save Classifier.
- void LoadClassifier ()

  Load Classifier.
- void ClearClassifier ()

  Clear classifier object.

Additional Inherited Members

5.22.1 Detailed Description

The RandomForest class.

5.22.2 Constructor & Destructor Documentation

5.22.2.1 RandomForest::RandomForest (string *clsFile,* CvRTParams *params,* unsigned int *numClasses*)

Class Constructor.

Parameters

clsFile	Path to the cassifier file
params	Structure for classifier parameters
numClasses	Total number of classes

5.22.3 Member Function Documentation

5.22.3.1 void RandomForest::Evaluate (Mat & *testData*, Mat & *possibleLabels*, Mat & *labelProbs*) `[virtual]`

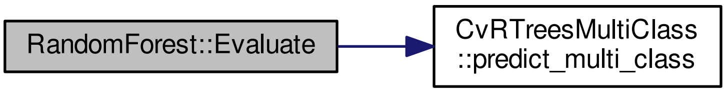
Predict class probabilities.

Parameters

testData	Test data, size of {number of samples x number of features}
possibleLabels	argmax< class probabilities> for each test sample
labelProbs	Class probabilities for each test sample

Implements Learning.

Here is the call graph for this function:



5.22.3.2 void RandomForest::Train (Mat & *trainData*, const Mat & *labels* = `Mat()`) `[virtual]`

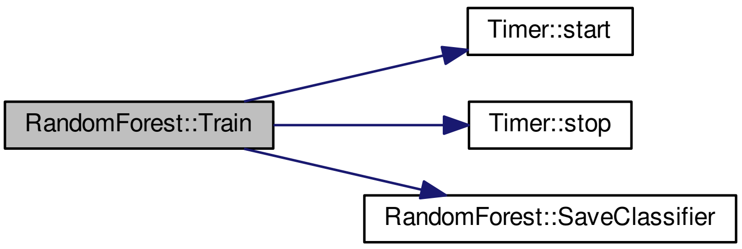
Train Classifier.

Parameters

trainData	Training data, size of {number of samples x number of features}
labels	Ground truth class labels, size of {1 x number of samples}

Implements Learning.

Here is the call graph for this function:



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

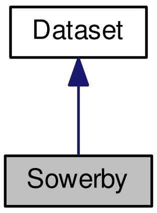
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Learning.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Learning.cpp

5.23 Sowerby Class Reference

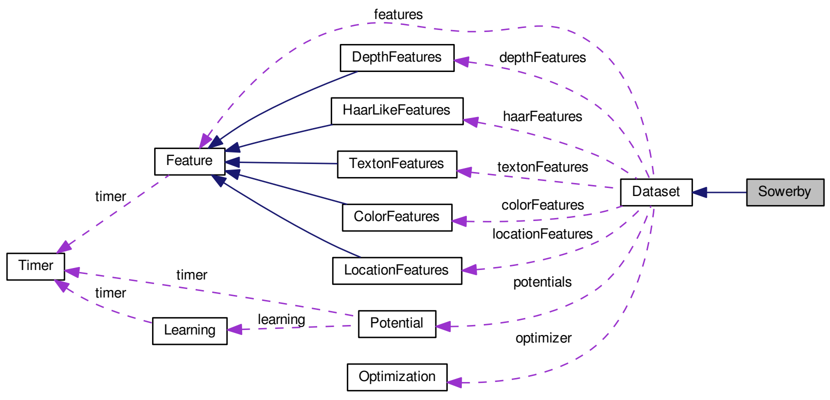
The Sowerby class.

`#include <Dataset.hpp>`

Inheritance diagram for Sowerby:



Collaboration diagram for Sowerby:



Public Member Functions

- Sowerby ()

  Class Constructor.

Additional Inherited Members

5.23.1 Detailed Description

The Sowerby class.

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

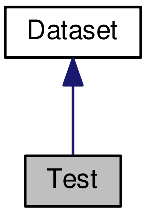
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Dataset.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Dataset.cpp

5.24 Test Class Reference

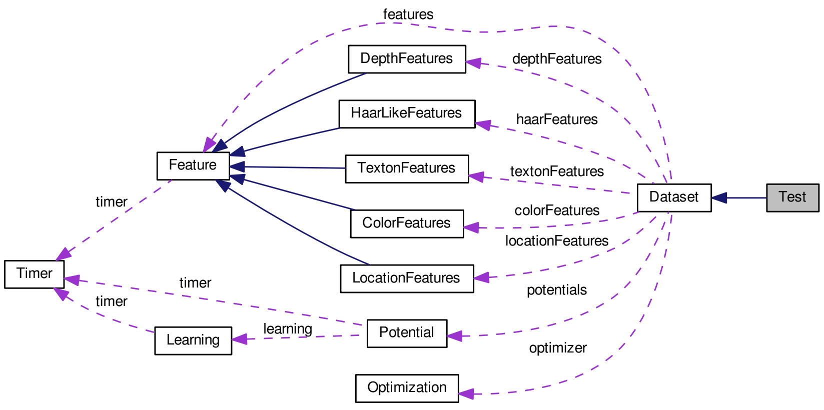
The Test class.

`#include <Dataset.hpp>`

Inheritance diagram for Test:



Collaboration diagram for Test:



Public Member Functions

- Test ()

  Class Constructor.

Additional Inherited Members

5.24.1 Detailed Description

The Test class.

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

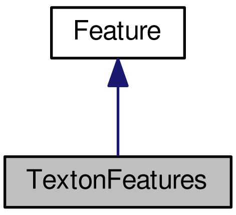
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Dataset.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Dataset.cpp

5.25 TextonFeatures Class Reference

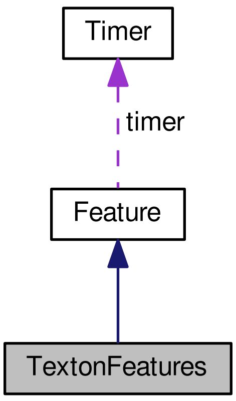
The TextonFeatures class.

```
#include <Feature.hpp>
```

Inheritance diagram for TextonFeatures:



Collaboration diagram for TextonFeatures:



Public Member Functions

- TextonFeatures (string TextonFolder, string ext, int numSubSample, double bandWidth)

  Class Constructor.
- void extractFeatures (Image ∗im, Mat &filterResponses)

  Extract features.

Additional Inherited Members

5.25.1 Detailed Description

The TextonFeatures class.

5.25.2 Constructor & Destructor Documentation

5.25.2.1 TextonFeatures::TextonFeatures (string *TextonFolder,* string *ext,* int *numSubSample,* double *bandWidth*)

Class Constructor.

Parameters

TextonFolder	folder to store features
ext	extension of feature files
numSubSample	sub sample size
bandWidth	bandwidth of texton kernels

5.25.3 Member Function Documentation

5.25.3.1 void TextonFeatures::extractFeatures (Image ∗ *im,* Mat & *features*) [virtual]

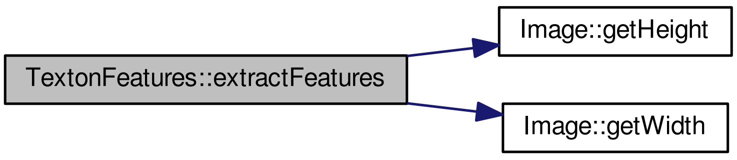
Extract features.

Parameters

im	image
features	feature matrix

Implements Feature.

Here is the call graph for this function:



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

- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Feature.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Feature.cpp

5.26 Timer Class Reference

The Timer class.

`#include <common.hpp>`

Public Member Functions

- void start ()

  start clock
- int stop (string msg="")

  stop clock
- void msg (int msecs, string msg)

  print out message

Static Public Member Functions

- static string formatted_time (int msecs)

  format time
- static int parse_formatted_time (string time_stream)

  parse formatted time
- static int num_digits (int num)

  number of digits
- static string to_string (int num)

  convert number to string

5.26.1 Detailed Description

The Timer class.

5.26.2 Member Function Documentation

5.26.2.1 static string Timer::formatted_time (int *msecs*) `[inline]`, `[static]`

format time

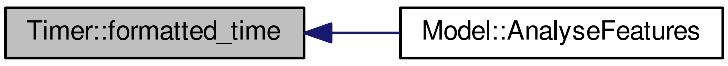
Parameters

msecs	miliseconds

Returns

formatted time

Here is the caller graph for this function:



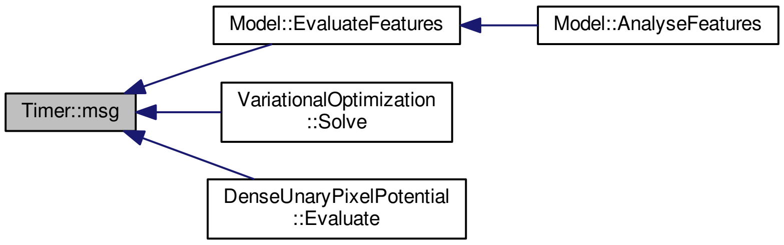
5.26.2.2 void Timer::msg (int *msecs,* string *msg*) `[inline]`

print out message

Parameters

msecs	miliseconds
msg	message

Here is the caller graph for this function:


Model::EvaluateFeatures
Model::AnalyseFeatures
Timer::msg
VariationalOptimization ::Solve
DenseUnaryPixelPotential ::Evaluate

5.26.2.3 static int Timer::num_digits (int *num*) [inline], [static]

number of digits

Parameters

num	number

Returns

total number of digits

5.26.2.4 static int Timer::parse_formatted_time (string *time_stream*) [inline], [static]

parse formatted time

Parameters

time_stream	time stream

Returns

time in miliseconds

5.26.2.5 int Timer::stop (string *msg* = "") [inline]

stop clock

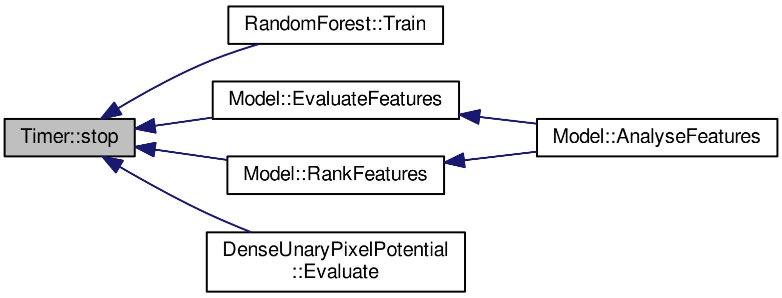
Parameters

msg	message to print out

Returns

time in miliseconds

Here is the caller graph for this function:



5.26.2.6 static string Timer::to_string (int *num*) `[inline]`, `[static]`

convert number to string

Parameters

num	number

Returns

string

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

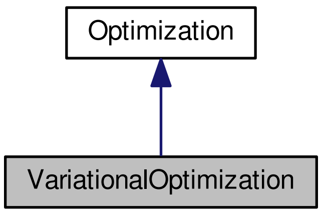
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/common.hpp

5.27 VariationalOptimization Class Reference

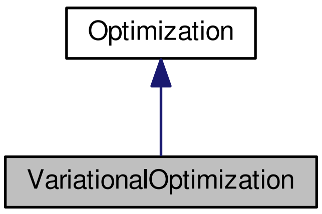
The VariationalOptimization class.

`#include <Optimization.hpp>`

Inheritance diagram for VariationalOptimization:



Collaboration diagram for VariationalOptimization:



Public Member Functions

- VariationalOptimization ()

    Class Constructor.
- int Solve (Dataset ∗dataset, vector< string > &imageList, int from, int to, bool printTime=true)

    optimize solution

5.27.1 Detailed Description

The VariationalOptimization class.

5.27.2 Member Function Documentation

5.27.2.1 int VariationalOptimization::Solve (**Dataset** ∗ *dataset,* vector< string > & *imageList,* int *from,* int *to,* bool *printTime* =`true`) `[virtual]`

optimize solution

Parameters

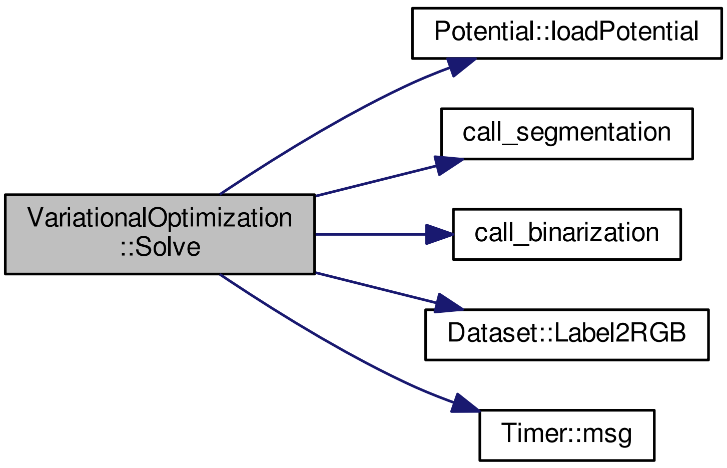
dataset	pointer to the current dataset
imageList	list of images to segment
from	start index
to	stop index, set to -1 to segment all images starting from "from"
printTime	flag to print out the estiamted time on output

Returns

estimated time per image in miliseconds

Implements Optimization.

Here is the call graph for this function:



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

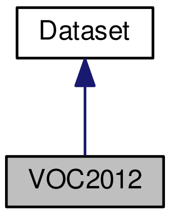
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Optimization.hpp
- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/src/Optimization.cpp

5.28 VOC2012 Class Reference

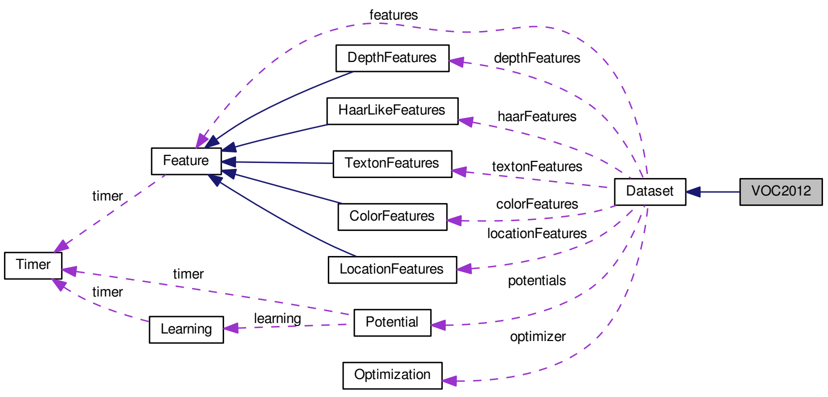
The VOC2012 class.

```
#include <Dataset.hpp>
```

Inheritance diagram for VOC2012:



Collaboration diagram for VOC2012:



Public Member Functions

- VOC2012 ()

  Class Constructor.
- void RGB2Label (Mat rgb, Mat &labels)

  Convert from RGB to Label.
- void Label2RGB (Mat labels, Mat &rgb)

  Convert from label to RGB.

Additional Inherited Members

5.28.1 Detailed Description

The VOC2012 class.

5.28.2 Member Function Documentation

5.28.2.1 void VOC2012::Label2RGB (Mat *labels,* Mat & *rgb*) [virtual]

Convert from label to RGB.

Parameters

labels	label matrix
rgb	RGB image

Reimplemented from Dataset.

5.28.2.2 void VOC2012::RGB2Label (Mat *rgb*, Mat & *labels*) `[virtual]`

Convert from RGB to Label.

Parameters

rgb	RGB image
labels	label matrix to store labels for each pixel

Reimplemented from Dataset.

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

- /usr/wiss/hazirbas/Work/Projects/thesis-hazibas/AFS/include/Dataset.hpp

Index