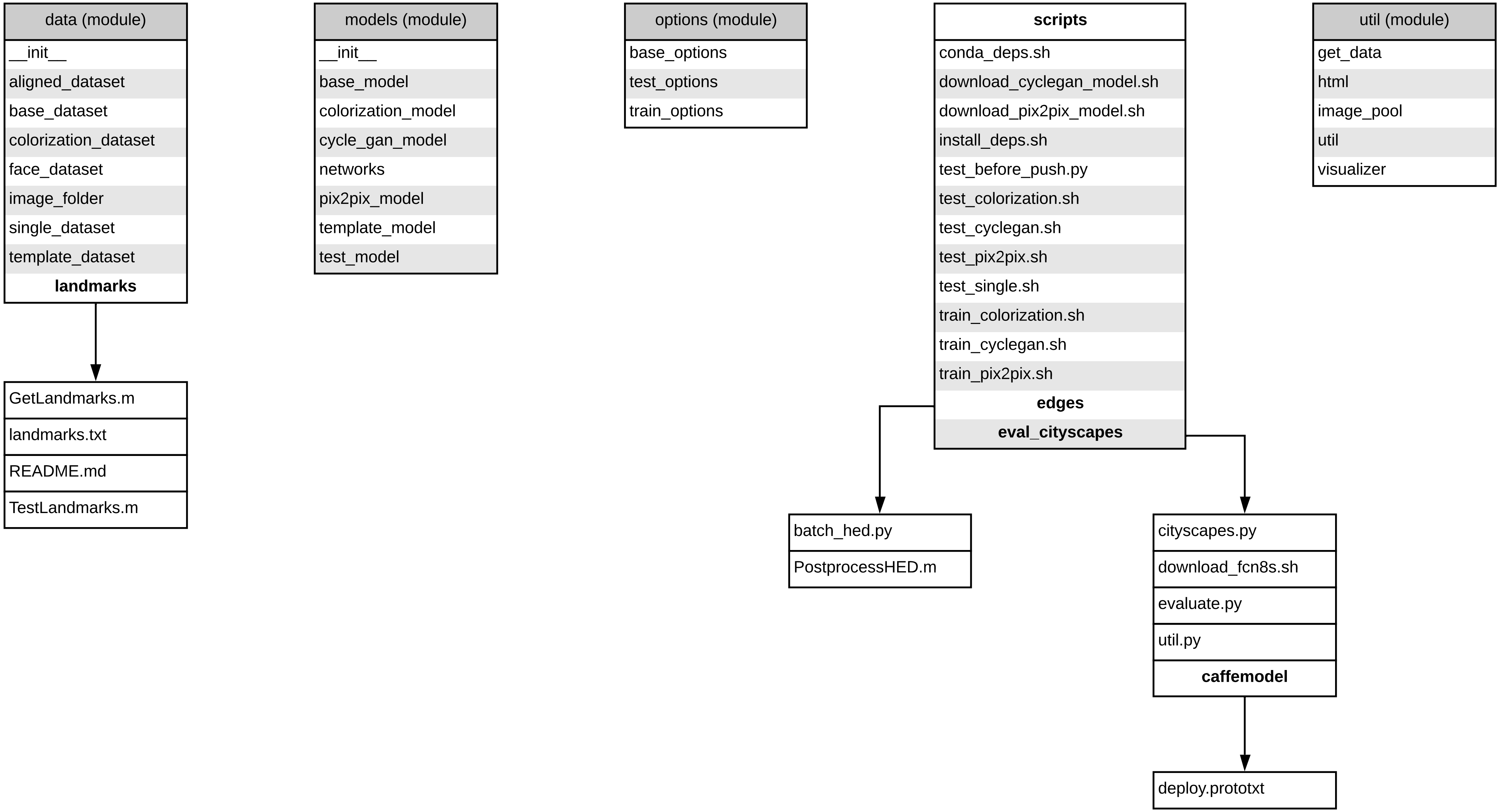
data (module)
__init__
aligned_dataset
base_dataset
colorization_dataset
face_dataset
image_folder
single_dataset
template_dataset
landmarks
GetLandmarks.m
landmarks.txt
README.md
TestLandmarks.m
models (module)
__init__
base_model
colorization_model
cycle_gan_model
networks
pix2pix_model
template_model
test_model
options (module)
base_options
test_options
train_options
scripts
conda_deps.sh
download_cyclegan_model.sh
download_pix2pix_model.sh
install_deps.sh
test_before_push.py
test_colorization.sh
test_cyclegan.sh
test_pix2pix.sh
test_single.sh
train_colorization.sh
train_cyclegan.sh
train_pix2pix.sh
edges
eval_cityscapes
batch_hed.py
PostprocessHED.m
cityscapes.py
download_fcn8s.sh
evaluate.py
util.py
caffemodel
deploy.prototxt
util (module)
get_data
html
image_pool
util
visualizer

models_part_1

base_model		
import	from collections import OrderedDict	
import	from . import networks	
class	BaseModel *(abstract class)*	
»	__init__	opt
» @stat	modify_commandline_options	parser, is_train
» @abs	set_input *(@abstractmethod)*	input
» @abs	forward	–
» @abs	optimize_parameters	–
»	setup	opt
»	eval	–
»	test	–
»	compute_visuals	–
»	get_image_paths	–
»	update_learning_rate	–
»	get_current_visuals	–
»	get_current_losses	–
»	save_networks	epoch
»	__patch_instance_norm_state_dict	state_dict, module, keys, i=0
»	load_networks	epoch
»	print_networks	verbose
»	set_require_grad	nets, requires_grad=False

__init__		
import	from .base_model import BaseModel	
def	find_model_using_name	model_name
def	get_option_setter	model_name
def	create_model	opt

colorization_model		
import	from .pix2pix_model import Pix2PixModel	
class	ColorizationModel	Pix2PixModel
» @stat	modify_commandline_options	parser, is_train=True
»	__init__	opt
»	lab2rgb	L, AB
»	compute_visuals	–

cycle_gan_model		
import	from util.image_pool import ImagePool	
import	from .base_model import BaseModel	
import	from . import networks	
class	CycleGANModel	BaseModel
» @stat	modify_commandline_options	parser, is_train=True
»	__init__	opt
»	setup_input	input
»	forward	–
»	backward_D_basic	netD, real, fake
»	backward_D_A	–
»	backward_D_B	–
»	backward_LD	–
»	backward_G	–
»	optimize_parameters	–

models_part_2

networks		
def	get_norm_layer	norm_type='instance'
def	get_scheduler	optimizer, opt
def	init_weights	net, init_type='normal', init_gain=0.02
def	init_net	net, inti_type='normal', init_gain=0.02, gpu_ids=[ ]
def	define_G	input_nc, output_nc, ngf, netG, norm='batch', use_dropout=False, init_type='normal', init_gain=0.02, gpu_ids=[ ]
class	LDNet	nn.Module
»	__init__	–
»	forward	input
def	define_LD	gpu_ids=[ ]
def	define_D	input_nc, ndf, netD, n_layers_D=3, norm='batch', init_type='normal', init_gain=0.02, gpu_ids=[ ]
class	GANLossLD	nn.Module
»	__init__	–
»	__call__	predictions, targets
class	GANLoss	nn.Module
»	__init__	gan_mode, target_real_label=1.0, target_fake_label=0.0
»	get_target_tensor	prediction, target_is_real
»	__call__	prediction, target_is_real
def	cal_gradient_penalty	netD, real_data, fake_data, device, type='mixed', constant=1.0, lambda_gp=10.0
class	ResnetGenerator	nn.Module
»	__init__	input_nc, output_nc, ngf=64, norm_layer=nn.BatchNorm2d, use_dropout=False, n_blocks=6, padding_type='reflect'
»	forward	input
class	ResnetBlock	nn.Module
»	__init__	dim, padding_type, norm_layer, use_dropout, use_bias
»	build_conv_block	dim, padding_type, norm_layer, use_dropout, use_bias
»	forward	x
class	UnetGenerator	nn.Module
»	__init__	input_nc, output_nc, num_downs, ngf=64, norm_layer=nn.BatchNorm2d, use_dropout=False
»	forward	input
class	UnetSkipConnectionBlock	nn.Module
»	__init__	outer_nc, inner_nc, input_nc=None, submodule=None, outermost=False, innermost=False, norm_layer=nn.BatchNorm2d, use_dropout=False
»	forward	x
class	NLayerDiscriminator	nn.Module
»	__init__	input_nc, ndf=64, n_layers=3, norm_layer=nn.BatchNorm2d
»	forward	input
class	PixelDiscriminator	nn.Module
»	__init__	input_nc, ndf=64, norm_layer=nn.BatchNorm2d
»	forward	input

pix2pix_model		
import	from . import networks	
class	Pix2PixModel	BaseModel
» @stat	modify_commandline_options	parser, is_train=True
»	__init__	opt
»	setup_input	input
»	forward	–
»	backward_D	–
»	backward_G	–
»	optimize_parameters	–

template_model		
import	from .base_model import BaseModel	
import	from . import networks	
class	TemplateModel	BaseModel
» @stat	modify_commandline_options	parser, is_train=True
»	__init__	opt
»	set_input	input
»	forward	–
»	backward	–
»	optimize_parameters	–

test_model		
import	from .base_model import BaseModel	
import	from . import networks	
class	TestModel	BaseModel
» @stat	modify_commandline_options	parser, is_train=True
»	__init__	opt
»	set_input	input
»	forward	–
»	optimize_parameters	–

options

base_options		
import	models	
import	data	
class	BaseOptions	
»	__init__	–
»	initialize	parser
»	gather_options	–
»	print_options	opt
»	parse	–

test_options		
import	from .base_options import BaseOptions	
class	TestOptions	BaseOptions
»	initialize	parser

train_options		
import	from .base_options import BaseOptions	
class	TrainOptions	BaseOptions
»	initialize	parser

data

__init__		
import	from .base_dataset import BaseDataset	
def	find_dataset_using_name	dataset_name
def	get_option_setter	dataset_name
def	create_dataset	opt
def	CustomDatasetDataLoader	–

aligned_dataset		
import	from data.base_dataset import BaseDataset, get_params, get_transform	
import	from data.image_folder import make_dataset	
class	AlignedDataset	BaseDataset
»	__init__	opt
»	__getitem__	index
»	__len__	–

base_dataset		
class	BaseDataset *(abstract class)*	data.Dataset
»	__init__	opt
» @stat	modify_commandline_options	parser, is_train
» @abs	__len__	–
» @abs	__getitem__	index
def	get_params	opt, size
def	get_transform	opt, params=None, grayscale=False, method=Image.BICUBIC, convert=True
def	__make_power_2	img, base, method=Image.BICUBIC
def	__scale_width	img, target_width, method=Image.BICUBIC
def	__crop	img, pos, size
def	__flip	img, flip
def	__print_size_warning	ow, oh, w, h

single_dataset		
import	from data.base_dataset import BaseDataset, get_transform	
import	data.image_folder import make_dataset	
class	SingleDataset	BaseDataset
»	__init__	opt
»	__getitem__	index
»	__len__	–

colorization_dataset		
import	from data.base_dataset import BaseDataset, get_transform	
import	from data.image_folder import make_dataset	
class	ColorizationDataset	BaseDataset
» @stat	modify_commandline_options	parser, is_train
»	__init__	opt
»	__getitem__	index
»	__len__	Type

face_dataset		
import	from data.base_dataset import BaseDataset	
import	from data.image_folder import make_dataset	
class	FaceDataset	BaseDataset
»	__init__	opt
»	get_transform_wide	centerCrop=False, resize=False
»	get_transform	–
»	__getitem__	index
»	ld	x, y
»	get_rand_upperleft	–
»	random_crop	img, rand
»	__len__	–

image_folder		
def	is_image_file	filename
def	make_dataset	dir, max_dataset_size=float("inf")
def	default_loader	path
class	ImageFolder	data.Dataset
»	__init__	root, transform=None, return_paths=False, loader=default_loader
»	__getitem__	index
»	__len__	–

template_dataset		
import	from data.base_dataset import BaseDataset, get_transform	
class	TemplateDataset	BaseDataset
» @stat	modify_commandline_options	parser, is_train
»	__getitem__	index
»	__len__	–

util

get_data		
class	GetData	
»	__init__	technique='cyclegan', verbose=True
»	_print	text
» @stat	_get_options	r
»	_present_options	–
»	_download_data	dataset_url, save_path
»	get	save_path, dataset=None

html		
class	HTML	
»	__init__	web_dir, title, refresh=0
»	get_image_dir	–
»	add_header	text
»	add_images	ims, txts, links, width=400
»	save	–

image_pool		
class	ImagePool	
»	__init__	pool_size
»	query	images

util		
def	tensor2im	input_image, imtype=np.uint8
def	diagnose_network	net, name='network'
def	save_image	image_numpy, image_path
def	print_numpy	x, val=True, shp=False
def	mkdirs	paths
def	mkdir	paths

visualizer		
def	save_images	webpage, visuals, image_path, aspect_ratio=1.0, width=256
class	Visualizer	
»	__init__	opt
»	reset	–
»	create_visdom_connections	–
»	display_current_results	visuals, epoch, save_results
»	plot_current_losses	epoch, counter_ratio, losses
»	print_current_losses	epoch, iters, losses, t_comp, t_data