models.dense_net.DenseNet

- m __init__(self, data_provider, growth_rate, depth, total_blocks, keep_prob, num_inter_threads, num_intra_threads, weight_decay, nesterov_momentum, model_type, dataset, should_save_logs, should_save_model, renew_logs=False, reduction=1.0, bc_mode=False, **kwargs)
- m _initialize_session(self)
- m _count_trainable_params(self)
- p save_path(self)
- p logs_path(self)
- p model_identifier(self)
- m save_model(self, global_step=None)
- m load_model(self)
- m log_loss_accuracy(self, loss, accuracy, epoch, prefix, should_print=True)
- m _define_inputs(self)
- m composite_function(self, _input, out_features, kernel_size=3)
- m bottleneck(self, _input, out_features)
- m add_internal_layer(self, _input, growth_rate)
- m add_block(self, _input, growth_rate, layers_per_block)
- m transition_layer(self, _input)
- m transition_layer_to_classes(self, _input)
- m conv2d(self, _input, out_features, kernel_size, strides=[1, 1, 1, 1], padding='SAME')
- m avg_pool(self, _input, k)
- m batch_norm(self, _input)
- m dropout(self, _input)
- m weight_variable_msra(self, shape, name)
- m weight_variable_xavier(self, shape, name)
- m bias_variable(self, shape, name='bias')
- m _build_graph(self)
- m train_all_epochs(self, train_params)
- m train_one_epoch(self, data, batch_size, learning_rate)
- m test(self, data, batch_size)

- f should_save_model
- f train_step
- f num_intra_threads
- f sess
- f data_shape
- f accuracy
- f total_blocks
- f nesterov_momentum
- f n_classes
- f is_training
- f _save_path
- f should_save_logs
- f growth_rate
- f num_inter_threads
- f saver
- f summary_writer
- f data_provider
- f keep_prob
- f bc_mode
- f images
- f layers_per_block
- f weight_decay
- f _logs_path
- f first_output_features
- f batches_step
- f model_type
- f renew_logs
- f dataset_name
- f cross_entropy
- f labels
- f depth
- f reduction
- f learning_rate