

ceras

Generated by Doxygen 1.9.1

1 Namespace Index	1
1.1 Namespace List	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Namespace Documentation	9
5.1 ceras Namespace Reference	9
5.1.1 Typedef Documentation	16
5.1.1.1 ada_delta	16
5.1.1.2 ada_grad	16
5.1.1.3 default_allocator	16
5.1.1.4 rms_prop	16
5.1.2 Function Documentation	16
5.1.2.1 abs() [1/2]	16
5.1.2.2 abs() [2/2]	17
5.1.2.3 abs_loss()	17
5.1.2.4 add()	17
5.1.2.5 amax()	17
5.1.2.6 amin()	17
5.1.2.7 as_tensor()	17
5.1.2.8 average_pooling_2d()	18
5.1.2.9 batch_normalization()	18
5.1.2.10 BatchNormalization()	18
5.1.2.11 clip() [1/2]	18
5.1.2.12 clip() [2/2]	18
5.1.2.13 concat() [1/2]	18
5.1.2.14 concat() [2/2]	19
5.1.2.15 concatenate() [1/3]	19
5.1.2.16 concatenate() [2/3]	19
5.1.2.17 concatenate() [3/3]	19
5.1.2.18 Conv2D()	19
5.1.2.19 conv2d()	20
5.1.2.20 copy() [1/2]	20
5.1.2.21 copy() [2/2]	20
5.1.2.22 cross_entropy()	20
5.1.2.23 cross_entropy_loss()	20
5.1.2.24 deep_copy()	21

5.1.2.25 Dense()	21
5.1.2.26 drop_out()	21
5.1.2.27 elementwise_divide()	21
5.1.2.28 elementwise_product() [1/2]	21
5.1.2.29 elementwise_product() [2/2]	21
5.1.2.30 elu()	22
5.1.2.31 empty()	22
5.1.2.32 exp()	22
5.1.2.33 exponential()	22
5.1.2.34 flatten()	22
5.1.2.35 gelu()	22
5.1.2.36 gemm() [1/2]	23
5.1.2.37 gemm() [2/2]	23
5.1.2.38 gemm_cpu()	23
5.1.2.39 get_default_session()	23
5.1.2.40 glorot_uniform()	23
5.1.2.41 hadamard_product() [1/2]	24
5.1.2.42 hadamard_product() [2/2]	24
5.1.2.43 hard_sigmoid()	24
5.1.2.44 has_inf()	24
5.1.2.45 has_nan()	24
5.1.2.46 hinge_loss()	24
5.1.2.47 identity()	25
5.1.2.48 img2col()	25
5.1.2.49 Input()	25
5.1.2.50 instance_normalization()	25
5.1.2.51 is_valid()	25
5.1.2.52 leaky_relu()	25
5.1.2.53 linspace()	26
5.1.2.54 load_tensor()	26
5.1.2.55 log()	26
5.1.2.56 mae()	26
5.1.2.57 max() [1/2]	26
5.1.2.58 max() [2/2]	27
5.1.2.59 max_pooling_2d()	27
5.1.2.60 maximum()	27
5.1.2.61 mean() [1/2]	27
5.1.2.62 mean() [2/2]	27
5.1.2.63 mean_absolute_error()	27
5.1.2.64 mean_reduce()	28
5.1.2.65 mean_squared_error()	28
5.1.2.66 mean_squared_logarithmic_error()	28

5.1.2.67 min() [1/2]	28
5.1.2.68 min() [2/2]	28
5.1.2.69 minus() [1/2]	28
5.1.2.70 minus() [2/2]	29
5.1.2.71 mse()	29
5.1.2.72 multiply() [1/2]	29
5.1.2.73 multiply() [2/2]	29
5.1.2.74 negative()	29
5.1.2.75 norm()	29
5.1.2.76 normalization_batch()	30
5.1.2.77 normalization_instance()	30
5.1.2.78 ones()	30
5.1.2.79 ones_like()	30
5.1.2.80 operator*() [1/4]	30
5.1.2.81 operator*() [2/4]	30
5.1.2.82 operator*() [3/4]	31
5.1.2.83 operator*() [4/4]	31
5.1.2.84 operator+() [1/4]	31
5.1.2.85 operator+() [2/4]	31
5.1.2.86 operator+() [3/4]	31
5.1.2.87 operator+() [4/4]	31
5.1.2.88 operator-() [1/4]	32
5.1.2.89 operator-() [2/4]	32
5.1.2.90 operator-() [3/4]	32
5.1.2.91 operator-() [4/4]	32
5.1.2.92 operator/()	32
5.1.2.93 operator<<()	32
5.1.2.94 operator==()	33
5.1.2.95 plus()	33
5.1.2.96 randn()	33
5.1.2.97 random()	33
5.1.2.98 random_like()	33
5.1.2.99 read_tensor()	34
5.1.2.100 reduce()	34
5.1.2.101 reduce_mean() [1/2]	34
5.1.2.102 reduce_mean() [2/2]	34
5.1.2.103 reduce_sum() [1/2]	34
5.1.2.104 reduce_sum() [2/2]	34
5.1.2.105 relu()	35
5.1.2.106 repeat()	35
5.1.2.107 repmat()	35
5.1.2.108 reshape() [1/2]	35

5.1.2.109 reshape() [2/2]	35
5.1.2.110 save_tensor()	35
5.1.2.111 selu()	36
5.1.2.112 sigmoid()	36
5.1.2.113 softmax() [1/2]	36
5.1.2.114 softmax() [2/2]	36
5.1.2.115 softplus()	36
5.1.2.116 softsign()	36
5.1.2.117 square()	37
5.1.2.118 squared_loss()	37
5.1.2.119 squeeze()	37
5.1.2.120 standard_deviation()	37
5.1.2.121 sum() [1/2]	37
5.1.2.122 sum() [2/2]	37
5.1.2.123 sum_reduce()	38
5.1.2.124 tanh()	38
5.1.2.125 transpose()	38
5.1.2.126 truncated_normal()	38
5.1.2.127 up_sampling_2d()	38
5.1.2.128 update_cuda_gemm_threshold()	38
5.1.2.129 variance()	39
5.1.2.130 write_tensor()	39
5.1.2.131 zeros()	39
5.1.2.132 zeros_like()	39
5.1.3 Variable Documentation	39
5.1.3.1 __version__	39
5.1.3.2 blas_mode	39
5.1.3.3 Constant	40
5.1.3.4 cuda_gemm_threshold	40
5.1.3.5 cuda_mode	40
5.1.3.6 debug_mode	40
5.1.3.7 eps	40
5.1.3.8 Expression	40
5.1.3.9 is_constant_v	40
5.1.3.10 is_operator_v	41
5.1.3.11 is_place_holder_v	41
5.1.3.12 is_tensor_v	41
5.1.3.13 is_value_v	41
5.1.3.14 is_variable_v	41
5.1.3.15 is_windows_platform	41
5.1.3.16 learning_phase	42
5.1.3.17 lstm	42

5.1.3.18 make_binary_operator	42
5.1.3.19 make_unary_operator	42
5.1.3.20 Operator	43
5.1.3.21 Place_Holder	43
5.1.3.22 random_generator	43
5.1.3.23 random_seed	43
5.1.3.24 Tensor	43
5.1.3.25 Value	43
5.1.3.26 Variable	43
5.1.3.27 version	43
5.1.3.28 visible_device	43
6 Class Documentation	45
6.1 ceras::adadelta< Loss, T > Struct Template Reference	45
6.1.1 Member Typedef Documentation	45
6.1.1.1 tensor_type	46
6.1.2 Constructor & Destructor Documentation	46
6.1.2.1 adadelta()	46
6.1.3 Member Function Documentation	46
6.1.3.1 forward()	46
6.1.4 Member Data Documentation	46
6.1.4.1 iterations_	46
6.1.4.2 learning_rate_	46
6.1.4.3 loss_	47
6.1.4.4 rho_	47
6.2 ceras::adagrad< Loss, T > Struct Template Reference	47
6.2.1 Member Typedef Documentation	47
6.2.1.1 tensor_type	48
6.2.2 Constructor & Destructor Documentation	48
6.2.2.1 adagrad()	48
6.2.3 Member Function Documentation	48
6.2.3.1 forward()	48
6.2.4 Member Data Documentation	48
6.2.4.1 decay_	48
6.2.4.2 iterations_	48
6.2.4.3 learning_rate_	49
6.2.4.4 loss_	49
6.3 ceras::adam< Loss, T > Struct Template Reference	49
6.3.1 Member Typedef Documentation	50
6.3.1.1 tensor_type	50
6.3.2 Constructor & Destructor Documentation	50
6.3.2.1 adam()	50

6.3.3 Member Function Documentation	50
6.3.3.1 forward()	50
6.3.4 Member Data Documentation	50
6.3.4.1 amsgrad_	50
6.3.4.2 beta_1_	51
6.3.4.3 beta_2_	51
6.3.4.4 iterations_	51
6.3.4.5 learning_rate_	51
6.3.4.6 loss_	51
6.4 ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action > Struct Template Reference	51
6.4.1 Member Typedef Documentation	52
6.4.1.1 tensor_type	52
6.4.2 Constructor & Destructor Documentation	52
6.4.2.1 binary_operator()	52
6.4.3 Member Function Documentation	53
6.4.3.1 backward()	53
6.4.3.2 forward()	53
6.4.3.3 reset_states()	53
6.4.4 Member Data Documentation	53
6.4.4.1 backward_action_	53
6.4.4.2 forward_action_	53
6.4.4.3 lhs_input_data_	54
6.4.4.4 lhs_op_	54
6.4.4.5 output_data_	54
6.4.4.6 reset_action_	54
6.4.4.7 rhs_input_data_	54
6.4.4.8 rhs_op_	55
6.5 ceras::constant< Tsor > Struct Template Reference	55
6.5.1 Member Function Documentation	55
6.5.1.1 backward()	55
6.5.1.2 forward()	55
6.5.1.3 shape()	56
6.5.2 Member Data Documentation	56
6.5.2.1 data_	56
6.6 ceras::gradient_descent< Loss, T > Struct Template Reference	56
6.6.1 Member Typedef Documentation	57
6.6.1.1 tensor_type	57
6.6.2 Constructor & Destructor Documentation	57
6.6.2.1 gradient_descent()	57
6.6.3 Member Function Documentation	57
6.6.3.1 forward()	57

6.6.4 Member Data Documentation	57
6.6.4.1 learning_rate_	57
6.6.4.2 loss_	58
6.6.4.3 momentum_	58
6.7 ceras::is_constant< T > Struct Template Reference	58
6.8 ceras::is_constant< constant< Tsor > > Struct Template Reference	58
6.9 ceras::is_operator< T > Struct Template Reference	59
6.10 ceras::is_operator< binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_↔ Action > > Struct Template Reference	59
6.11 ceras::is_operator< unary_operator< Operator, Forward_Action, Backward_Action > > Struct Tem- plate Reference	59
6.12 ceras::is_place_holder< T > Struct Template Reference	60
6.13 ceras::is_place_holder< place_holder< Tsor > > Struct Template Reference	60
6.14 ceras::is_tensor< T > Struct Template Reference	60
6.15 ceras::is_tensor< tensor< T, A > > Struct Template Reference	61
6.16 ceras::is_value< T > Struct Template Reference	61
6.17 ceras::is_value< value< T > > Struct Template Reference	61
6.18 ceras::is_variable< T > Struct Template Reference	62
6.19 ceras::is_variable< variable< Tsor > > Struct Template Reference	62
6.20 ceras::place_holder< Tsor > Struct Template Reference	62
6.20.1 Member Typedef Documentation	63
6.20.1.1 tensor_type	63
6.20.2 Constructor & Destructor Documentation	63
6.20.2.1 place_holder() [1/4]	63
6.20.2.2 place_holder() [2/4]	63
6.20.2.3 place_holder() [3/4]	63
6.20.2.4 place_holder() [4/4]	64
6.20.3 Member Function Documentation	64
6.20.3.1 backward()	64
6.20.3.2 bind()	64
6.20.3.3 forward()	64
6.20.3.4 operator=() [1/2]	64
6.20.3.5 operator=() [2/2]	64
6.20.3.6 reset()	65
6.21 ceras::place_holder_state< Tsor > Struct Template Reference	65
6.21.1 Member Data Documentation	65
6.21.1.1 data_	65
6.21.1.2 shape_hint_	65
6.22 ceras::rmsprop< Loss, T > Struct Template Reference	65
6.22.1 Member Typedef Documentation	66
6.22.1.1 tensor_type	66
6.22.2 Constructor & Destructor Documentation	66
6.22.2.1 rmsprop()	66

6.22.3 Member Function Documentation	66
6.22.3.1 forward()	67
6.22.4 Member Data Documentation	67
6.22.4.1 decay_	67
6.22.4.2 iterations_	67
6.22.4.3 learning_rate_	67
6.22.4.4 loss_	67
6.22.4.5 rho_	67
6.23 ceras::session< Tsor > Struct Template Reference	68
6.23.1 Member Typedef Documentation	68
6.23.1.1 place_holder_type	68
6.23.1.2 variable_state_type	69
6.23.1.3 variable_type	69
6.23.2 Constructor & Destructor Documentation	69
6.23.2.1 session() [1/3]	69
6.23.2.2 session() [2/3]	69
6.23.2.3 session() [3/3]	69
6.23.2.4 ~session()	69
6.23.3 Member Function Documentation	70
6.23.3.1 bind()	70
6.23.3.2 deserialize()	70
6.23.3.3 operator=() [1/2]	70
6.23.3.4 operator=() [2/2]	70
6.23.3.5 rebind()	70
6.23.3.6 remember()	71
6.23.3.7 restore()	71
6.23.3.8 run()	71
6.23.3.9 save()	71
6.23.3.10 serialize()	71
6.23.3.11 tap()	71
6.23.4 Member Data Documentation	72
6.23.4.1 place_holders_	72
6.23.4.2 variables_	72
6.24 ceras::sgd< Loss, T > Struct Template Reference	72
6.24.1 Member Typedef Documentation	73
6.24.1.1 tensor_type	73
6.24.2 Constructor & Destructor Documentation	73
6.24.2.1 sgd()	73
6.24.3 Member Function Documentation	73
6.24.3.1 forward()	73
6.24.4 Member Data Documentation	73
6.24.4.1 decay_	74

6.24.4.2 iterations_	74
6.24.4.3 learning_rate_	74
6.24.4.4 loss_	74
6.24.4.5 momentum_	74
6.24.4.6 nesterov_	74
6.25 ceras::tensor< T, Allocator > Struct Template Reference	75
6.25.1 Member Typedef Documentation	76
6.25.1.1 allocator	76
6.25.1.2 self_type	76
6.25.1.3 shared_vector	76
6.25.1.4 value_type	77
6.25.1.5 vector_type	77
6.25.2 Constructor & Destructor Documentation	77
6.25.2.1 tensor() [1/7]	77
6.25.2.2 tensor() [2/7]	77
6.25.2.3 tensor() [3/7]	77
6.25.2.4 tensor() [4/7]	77
6.25.2.5 tensor() [5/7]	78
6.25.2.6 tensor() [6/7]	78
6.25.2.7 tensor() [7/7]	78
6.25.3 Member Function Documentation	78
6.25.3.1 begin() [1/2]	78
6.25.3.2 begin() [2/2]	78
6.25.3.3 cbegin()	78
6.25.3.4 cend()	79
6.25.3.5 copy()	79
6.25.3.6 creep_to()	79
6.25.3.7 data() [1/2]	79
6.25.3.8 data() [2/2]	79
6.25.3.9 deep_copy() [1/2]	79
6.25.3.10 deep_copy() [2/2]	80
6.25.3.11 empty()	80
6.25.3.12 end() [1/2]	80
6.25.3.13 end() [2/2]	80
6.25.3.14 map()	80
6.25.3.15 ndim()	80
6.25.3.16 operator*=() [1/2]	81
6.25.3.17 operator*=() [2/2]	81
6.25.3.18 operator+=() [1/2]	81
6.25.3.19 operator+=() [2/2]	81
6.25.3.20 operator-()	81
6.25.3.21 operator-=() [1/2]	81

6.25.3.22 operator==() [2/2]	82
6.25.3.23 operator/=() [1/2]	82
6.25.3.24 operator/=() [2/2]	82
6.25.3.25 operator=() [1/2]	82
6.25.3.26 operator=() [2/2]	82
6.25.3.27 operator[] () [1/2]	82
6.25.3.28 operator[] () [2/2]	83
6.25.3.29 reset ()	83
6.25.3.30 reshape ()	83
6.25.3.31 resize ()	83
6.25.3.32 shape ()	83
6.25.3.33 shrink_to ()	83
6.25.3.34 size ()	84
6.25.3.35 slice ()	84
6.25.4 Member Data Documentation	84
6.25.4.1 memory_offset_	84
6.25.4.2 shape_	84
6.25.4.3 vector_	84
6.26 ceras::tensor_deduction< L, R > Struct Template Reference	84
6.26.1 Member Typedef Documentation	85
6.26.1.1 op_type	85
6.26.1.2 tensor_type	85
6.27 ceras::unary_operator< Operator, Forward_Action, Backward_Action > Struct Template Reference	85
6.27.1 Constructor & Destructor Documentation	86
6.27.1.1 unary_operator ()	86
6.27.2 Member Function Documentation	86
6.27.2.1 backward ()	86
6.27.2.2 forward ()	86
6.27.2.3 reset_states ()	86
6.27.3 Member Data Documentation	87
6.27.3.1 backward_action_	87
6.27.3.2 forward_action_	87
6.27.3.3 input_data_	87
6.27.3.4 op_	87
6.27.3.5 output_data_	87
6.27.3.6 reset_action_	87
6.27.3.7 tensor_type	88
6.28 ceras::value< T > Struct Template Reference	88
6.28.1 Member Typedef Documentation	88
6.28.1.1 value_type	88
6.28.2 Constructor & Destructor Documentation	89
6.28.2.1 value () [1/4]	89

6.28.2.2 value() [2/4]	89
6.28.2.3 value() [3/4]	89
6.28.2.4 value() [4/4]	89
6.28.3 Member Function Documentation	89
6.28.3.1 backward()	89
6.28.3.2 forward()	90
6.28.3.3 operator=() [1/2]	90
6.28.3.4 operator=() [2/2]	90
6.28.4 Member Data Documentation	90
6.28.4.1 data_	90
6.29 ceras::variable< Tsor > Struct Template Reference	90
6.29.1 Member Typedef Documentation	91
6.29.1.1 tensor_type	91
6.29.2 Constructor & Destructor Documentation	91
6.29.2.1 variable() [1/4]	91
6.29.2.2 variable() [2/4]	92
6.29.2.3 variable() [3/4]	92
6.29.2.4 variable() [4/4]	92
6.29.3 Member Function Documentation	92
6.29.3.1 backward()	92
6.29.3.2 data() [1/2]	92
6.29.3.3 data() [2/2]	92
6.29.3.4 forward()	93
6.29.3.5 gradient() [1/2]	93
6.29.3.6 gradient() [2/2]	93
6.29.3.7 operator=() [1/2]	93
6.29.3.8 operator=() [2/2]	93
6.29.3.9 reset()	93
6.29.3.10 reset_states()	93
6.29.3.11 shape()	94
6.29.4 Member Data Documentation	94
6.29.4.1 state_	94
6.29.4.2 stateful_	94
6.29.4.3 trainable_	94
6.30 ceras::variable_state< Tsor > Struct Template Reference	94
6.30.1 Member Data Documentation	94
6.30.1.1 data_	95
6.30.1.2 gradient_	95
6.31 ceras::view_2d< T > Struct Template Reference	95
6.31.1 Constructor & Destructor Documentation	95
6.31.1.1 view_2d() [1/3]	96
6.31.1.2 view_2d() [2/3]	96

6.31.1.3 view_2d() [3/3]	96
6.31.2 Member Function Documentation	96
6.31.2.1 data() [1/2]	96
6.31.2.2 data() [2/2]	96
6.31.2.3 operator[]() [1/2]	97
6.31.2.4 operator[]() [2/2]	97
6.31.2.5 shape()	97
6.31.2.6 size()	97
6.31.3 Member Data Documentation	97
6.31.3.1 col_	97
6.31.3.2 data_	97
6.31.3.3 row_	98
6.31.3.4 transposed_	98
6.32 ceras::view_3d< T > Struct Template Reference	98
6.32.1 Constructor & Destructor Documentation	98
6.32.1.1 view_3d()	98
6.32.2 Member Function Documentation	99
6.32.2.1 operator[]() [1/2]	99
6.32.2.2 operator[]() [2/2]	99
6.32.3 Member Data Documentation	99
6.32.3.1 channel_	99
6.32.3.2 col_	99
6.32.3.3 data_	99
6.32.3.4 row_	100
6.33 ceras::view_4d< T > Struct Template Reference	100
6.33.1 Detailed Description	100
6.33.2 Constructor & Destructor Documentation	100
6.33.2.1 view_4d()	101
6.33.3 Member Function Documentation	102
6.33.3.1 operator[]() [1/2]	102
6.33.3.2 operator[]() [2/2]	102
6.33.4 Member Data Documentation	103
6.33.4.1 batch_size_	103
6.33.4.2 channel_	103
6.33.4.3 col_	103
6.33.4.4 data_	103
6.33.4.5 row_	103
7 File Documentation	105
7.1 /data/structured_folders/workspace/github.repo/ceras/include/activation.hpp File Reference	105
7.2 /data/structured_folders/workspace/github.repo/ceras/include/ceras.hpp File Reference	106
7.3 /data/structured_folders/workspace/github.repo/ceras/include/config.hpp File Reference	106

7.4 /data/structured_folders/workspace/github.repo/ceras/include/constant.hpp File Reference	107
7.5 /data/structured_folders/workspace/github.repo/ceras/include/includes.hpp File Reference	107
7.5.1 Macro Definition Documentation	108
7.5.1.1 STB_IMAGE_IMPLEMENTATION	108
7.5.1.2 STB_IMAGE_RESIZE_IMPLEMENTATION	108
7.5.1.3 STB_IMAGE_WRITE_IMPLEMENTATION	108
7.6 /data/structured_folders/workspace/github.repo/ceras/include/keras.hpp File Reference	109
7.7 /data/structured_folders/workspace/github.repo/ceras/include/layer.hpp File Reference	109
7.8 /data/structured_folders/workspace/github.repo/ceras/include/loss.hpp File Reference	109
7.9 /data/structured_folders/workspace/github.repo/ceras/include/operation.hpp File Reference	110
7.10 /data/structured_folders/workspace/github.repo/ceras/include/optimizer.hpp File Reference	112
7.11 /data/structured_folders/workspace/github.repo/ceras/include/place_holder.hpp File Reference	113
7.12 /data/structured_folders/workspace/github.repo/ceras/include/recurrent_operation.hpp File Reference	114
7.13 /data/structured_folders/workspace/github.repo/ceras/include/session.hpp File Reference	114
7.14 /data/structured_folders/workspace/github.repo/ceras/include/tensor.hpp File Reference	115
7.15 /data/structured_folders/workspace/github.repo/ceras/include/value.hpp File Reference	118
7.16 /data/structured_folders/workspace/github.repo/ceras/include/variable.hpp File Reference	119
Index	121

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

ceras	9
---------------------------------	---

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

ceras::constant< Tzor >	55
enable_id	
ceras::adadelta< Loss, T >	45
ceras::adagrad< Loss, T >	47
ceras::adam< Loss, T >	49
ceras::binary_operator< Lhs_Operator, Rhc_Operator, Forward_Action, Backward_Action >	51
ceras::gradient_descent< Loss, T >	56
ceras::place_holder< Tzor >	62
ceras::rmsprop< Loss, T >	65
ceras::sgd< Loss, T >	72
ceras::tensor< T, Allocator >	75
ceras::unary_operator< Operator, Forward_Action, Backward_Action >	85
ceras::variable< Tzor >	90
enable_shared	
ceras::adadelta< Loss, T >	45
ceras::adagrad< Loss, T >	47
ceras::adam< Loss, T >	49
ceras::gradient_descent< Loss, T >	56
ceras::rmsprop< Loss, T >	65
ceras::sgd< Loss, T >	72
enable_shared_state	
ceras::place_holder< Tzor >	62
std::false_type	
ceras::is_constant< T >	58
ceras::is_operator< T >	59
ceras::is_place_holder< T >	60
ceras::is_tensor< T >	60
ceras::is_value< T >	61
ceras::is_variable< T >	62
ceras::place_holder_state< Tzor >	65
ceras::session< Tzor >	68
ceras::tensor_deduction< L, R >	84
ceras::tensor_deduction< Lhs_Operator, Rhc_Operator >	84
std::true_type	
ceras::is_constant< constant< Tzor > >	58

ceras::is_operator< binary_operator< Lhs_Operator, Rhc_Operator, Forward_Action, Backward_↔ Action > >	59
ceras::is_operator< unary_operator< Operator, Forward_Action, Backward_Action > >	59
ceras::is_place_holder< place_holder< Tsor > >	60
ceras::is_tensor< tensor< T, A > >	61
ceras::is_value< value< T > >	61
ceras::is_variable< variable< Tsor > >	62
ceras::value< T >	88
ceras::variable_state< Tsor >	94
ceras::view_2d< T >	95
ceras::view_3d< T >	98
ceras::view_4d< T >	100

Chapter 3

Class Index

3.1 Class List

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

ceras::adadelta< Loss, T >	45
ceras::adagrad< Loss, T >	47
ceras::adam< Loss, T >	49
ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >	51
ceras::constant< Tsor >	55
ceras::gradient_descent< Loss, T >	56
ceras::is_constant< T >	58
ceras::is_constant< constant< Tsor > >	58
ceras::is_operator< T >	59
ceras::is_operator< binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action > >	59
ceras::is_operator< unary_operator< Operator, Forward_Action, Backward_Action > >	59
ceras::is_place_holder< T >	60
ceras::is_place_holder< place_holder< Tsor > >	60
ceras::is_tensor< T >	60
ceras::is_tensor< tensor< T, A > >	61
ceras::is_value< T >	61
ceras::is_value< value< T > >	61
ceras::is_variable< T >	62
ceras::is_variable< variable< Tsor > >	62
ceras::place_holder< Tsor >	62
ceras::place_holder_state< Tsor >	65
ceras::rmsprop< Loss, T >	65
ceras::session< Tsor >	68
ceras::sgd< Loss, T >	72
ceras::tensor< T, Allocator >	75
ceras::tensor_deduction< L, R >	84
ceras::unary_operator< Operator, Forward_Action, Backward_Action >	85
ceras::value< T >	88
ceras::variable< Tsor >	90
ceras::variable_state< Tsor >	94
ceras::view_2d< T >	95
ceras::view_3d< T >	98
ceras::view_4d< T >	100

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

/data/structured_folders/workspace/github.repo/ceras/include/activation.hpp	105
/data/structured_folders/workspace/github.repo/ceras/include/ceras.hpp	106
/data/structured_folders/workspace/github.repo/ceras/include/config.hpp	106
/data/structured_folders/workspace/github.repo/ceras/include/constant.hpp	107
/data/structured_folders/workspace/github.repo/ceras/include/includes.hpp	107
/data/structured_folders/workspace/github.repo/ceras/include/layer.hpp	109
/data/structured_folders/workspace/github.repo/ceras/include/keras.hpp	109
/data/structured_folders/workspace/github.repo/ceras/include/loss.hpp	109
/data/structured_folders/workspace/github.repo/ceras/include/operation.hpp	110
/data/structured_folders/workspace/github.repo/ceras/include/optimizer.hpp	112
/data/structured_folders/workspace/github.repo/ceras/include/place_holder.hpp	113
/data/structured_folders/workspace/github.repo/ceras/include/recurrent_operation.hpp	114
/data/structured_folders/workspace/github.repo/ceras/include/session.hpp	114
/data/structured_folders/workspace/github.repo/ceras/include/tensor.hpp	115
/data/structured_folders/workspace/github.repo/ceras/include/value.hpp	118
/data/structured_folders/workspace/github.repo/ceras/include/variable.hpp	119

Chapter 5

Namespace Documentation

5.1 ceras Namespace Reference

Classes

- struct [constant](#)
- struct [is_constant](#)
- struct [is_constant< constant< Tsor > >](#)
- struct [unary_operator](#)
- struct [binary_operator](#)
- struct [is_operator](#)
- struct [is_operator< binary_operator< Lhs_Operator, Rhc_Operator, Forward_Action, Backward_Action > >](#)
- struct [is_operator< unary_operator< Operator, Forward_Action, Backward_Action > >](#)
- struct [sgd](#)
- struct [adagrad](#)
- struct [rmsprop](#)
- struct [adadelat](#)
- struct [adam](#)
- struct [gradient_descent](#)
- struct [place_holder_state](#)
- struct [place_holder](#)
- struct [is_place_holder](#)
- struct [is_place_holder< place_holder< Tsor > >](#)
- struct [session](#)
- struct [tensor](#)
- struct [is_tensor](#)
- struct [is_tensor< tensor< T, A > >](#)
- struct [view_2d](#)
- struct [view_3d](#)
- struct [view_4d](#)
- struct [value](#)
- struct [is_value](#)
- struct [is_value< value< T > >](#)
- struct [tensor_deduction](#)
- struct [variable_state](#)
- struct [variable](#)
- struct [is_variable](#)
- struct [is_variable< variable< Tsor > >](#)

Typedefs

- `template<typename Loss , typename T >`
using `ada_grad` = `adagrad`< Loss, T >
- `template<typename Loss , typename T >`
using `rms_prop` = `rmsprop`< Loss, T >
- `template<typename Loss , typename T >`
using `ada_delta` = `adadelta`< Loss, T >
- `template<typename T >`
using `default_allocator` = `std::allocator`< T >

Functions

- `template<Expression Ex>`
constexpr auto `softmax` (Ex const &ex) noexcept
- `template<Expression Ex>`
auto `selu` (Ex const &ex) noexcept
- `template<Expression Ex>`
auto `softplus` (Ex const &ex) noexcept
- `template<Expression Ex>`
auto `softsign` (Ex const &ex) noexcept
- `template<Expression Ex>`
auto `sigmoid` (Ex const &ex) noexcept
- `template<Expression Ex>`
auto `tanh` (Ex const &ex) noexcept
- `template<Expression Ex>`
auto `relu` (Ex const &ex) noexcept
- `template<typename T >`
requires `std::floating_point`< T > auto `leaky_relu` (T const factor) noexcept
- `template<typename T >`
requires `std::floating_point`< T > auto `elu` (T const alpha) noexcept
- `template<Expression Ex>`
auto `exponential` (Ex const &ex) noexcept
- `template<Expression Ex>`
auto `hard_sigmoid` (Ex const &ex) noexcept
- `template<Expression Ex>`
auto `gelu` (Ex const &ex) noexcept
- auto `Input` ()
- auto `Conv2D` (unsigned long output_channels, `std::vector`< unsigned long > const &kernel_size, `std::vector`< unsigned long > const &input_shape, `std::string` const &padding="valid", `std::vector`< unsigned long > const &strides={1, 1})
- auto `Dense` (unsigned long output_size, unsigned long input_size)
- auto `BatchNormalization` (`std::vector`< unsigned long > const &shape, float threshold=0.95f)
- `template<Expression Lhs_Expression, Expression Rh_Expression>`
constexpr auto `mean_squared_logarithmic_error` (Lhs_Expression const &lhs_ex, Rh_Expression const &rhs_ex) noexcept
- `template<Expression Lhs_Expression, Expression Rh_Expression>`
constexpr auto `squared_loss` (Lhs_Expression const &lhs_ex, Rh_Expression const &rhs_ex) noexcept
- `template<Expression Lhs_Expression, Expression Rh_Expression>`
constexpr auto `mean_squared_error` (Lhs_Expression const &lhs_ex, Rh_Expression const &rhs_ex) noexcept
- `template<Expression Lhs_Expression, Expression Rh_Expression>`
constexpr auto `mse` (Lhs_Expression const &lhs_ex, Rh_Expression const &rhs_ex) noexcept
- `template<Expression Lhs_Expression, Expression Rh_Expression>`
constexpr auto `abs_loss` (Lhs_Expression const &lhs_ex, Rh_Expression const &rhs_ex) noexcept

- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto mean_absolute_error (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto mae (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto cross_entropy (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto cross_entropy_loss (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto hinge_loss (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto plus (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto operator+ (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`auto operator* (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Ex>`
`constexpr auto log (Ex const &ex) noexcept`
- `template<Expression Ex>`
`constexpr auto negative (Ex const &ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto elementwise_product (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto hadamard_product (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Ex>`
`constexpr auto sum_reduce (Ex const &ex) noexcept`
- `template<Expression Ex>`
`constexpr auto reduce_sum (Ex const &ex) noexcept`
- `template<Expression Ex>`
`constexpr auto mean_reduce (Ex const &ex) noexcept`
- `template<Expression Ex>`
`constexpr auto reduce_mean (Ex const &ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto minus (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto operator- (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Ex>`
`constexpr auto square (Ex const &ex) noexcept`
- `template<Expression Ex>`
`constexpr auto abs (Ex const &ex) noexcept`
- `template<Expression Ex>`
`constexpr auto exp (Ex const &ex) noexcept`
- `template<typename Float >`
`requires std::floating_point< Float > constexpr auto clip (Float lower, Float upper=std::numeric_limits< Float >::max()) noexcept`
- `auto reshape (std::vector< unsigned long > const &new_shape, bool include_batch_flag=true) noexcept`
- `template<Expression Ex>`
`constexpr auto flatten (Ex const &ex) noexcept`
- `template<Expression Ex>`
`constexpr auto identity (Ex const &ex) noexcept`
- `template<Expression Ex>`
`auto transpose (Ex const &ex) noexcept`

- auto [img2col](#) (unsigned long const row_kernel, unsigned long col_kernel=-1, unsigned long const row_↵padding=0, unsigned long col_padding=0, unsigned long const row_stride=1, unsigned long const col_↵stride=1, unsigned long const row_dilation=1, unsigned long const col_dilation=1) noexcept
- auto [conv2d](#) (unsigned long row_input, unsigned long col_input, unsigned long const row_stride=1, unsigned long const col_stride=1, unsigned long const row_dilation=1, unsigned long const col_dilation=1, std::string const &padding="valid") noexcept
- template<typename T >
requires std::floating_point< T > auto [drop_out](#) (T const factor) noexcept
- auto [max_pooling_2d](#) (unsigned long stride) noexcept
- auto [average_pooling_2d](#) (unsigned long stride) noexcept
- auto [up_sampling_2d](#) (unsigned long stride) noexcept
- template<typename T = double>
requires std::floating_point< T > auto [normalization_batch](#) (T const momentum=0.98) noexcept
- template<typename T >
requires std::floating_point< T > auto [batch_normalization](#) (T const momentum=0.98) noexcept
- template<typename T = double>
requires std::floating_point< T > auto [normalization_instance](#) (T const momentum=0.98) noexcept
- template<typename T >
requires std::floating_point< T > auto [instance_normalization](#) (T const momentum=0.98) noexcept
- template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto [concatenate](#) (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept
- auto [concatenate](#) (unsigned long axe=-1)
- template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto [concat](#) (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept
- auto [concat](#) (unsigned long axe=-1)
- template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto [maximum](#) (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept
- template<Expression Lhs_Expression, Variable Rhs_Variable>
constexpr auto [copy](#) (Lhs_Expression const &lhs_ex, Rhs_Variable const &rhs_va) noexcept
- template<Tensor Tsr>
std::reference_wrapper< [session](#)< Tsr > > [get_default_session](#) ()
- template<typename T, typename A = default_allocator<T>>
constexpr [tensor](#)< T, A > [as_tensor](#) (T val) noexcept
- template<Tensor Tsr, typename CharT, typename Traits >
std::basic_ostream< CharT, Traits > & [operator<<](#) (std::basic_ostream< CharT, Traits > &os_, Tsr const &tsr)
- template<typename T >
requires std::floating_point< T > void [gemm_cpu](#) (T const *A, bool a_transposed, T const *B, bool b_↵transposed, unsigned long m, unsigned long n, unsigned long k, T *C)
- void [update_cuda_gemm_threshold](#) ()
- template<typename T >
requires std::floating_point< T > void [gemm](#) (T const *A, bool a_transposed, T const *B, bool b_transposed, unsigned long m, unsigned long n, unsigned long k, T *C)
- template<typename T >
requires std::floating_point< T > void [gemm](#) ([view_2d](#)< T > const &x, [view_2d](#)< T > const &y, [view_2d](#)< T > &ans)
- template<Tensor Tsr>
Tsr [add](#) (Tsr const &lhs, Tsr const &rhs) noexcept
- template<Tensor Tsr>
Tsr [operator+](#) (Tsr const &lhs, Tsr const &rhs) noexcept
- template<Tensor Tsr>
Tsr [operator+](#) (typename Tsr::value_type const &lhs, Tsr const &rhs) noexcept
- template<Tensor Tsr>
Tsr [operator+](#) (Tsr const &lhs, typename Tsr::value_type const &rhs) noexcept
- template<Tensor Tsr>
Tsr [minus](#) (Tsr const &lhs, Tsr const &rhs) noexcept

- `template<Tensor T>`
`T` [operator-](#) (`T` const &`lhs`, `T` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [operator-](#) (`typename T::value_type` const &`lhs`, `T` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [operator-](#) (`T` const &`lhs`, `typename T::value_type` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [operator*](#) (`typename T::value_type` const &`lhs`, `T` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [operator*](#) (`T` const &`lhs`, `typename T::value_type` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [operator/](#) (`T` const &`lhs`, `typename T::value_type` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [reshape](#) (`T` const &`ts`, `std::vector< unsigned long >` const &`new_shape`)
- `template<Tensor T>`
`void` [multiply](#) (`T` const &`lhs`, `T` const &`rhs`, `T` &`ans`) noexcept
- `template<Tensor T>`
`T` [multiply](#) (`T` const &`lhs`, `T` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [operator*](#) (`T` const &`lhs`, `T` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [elementwise_product](#) (`T` const &`lhs`, `T` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [hadamard_product](#) (`T` const &`lhs`, `T` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [elementwise_divide](#) (`T` const &`lhs`, `T` const &`rhs`) noexcept
- `template<Tensor T>`
`T` [repeat](#) (`T` const &`tsor`, `unsigned long` `n`)
- `template<Tensor T>`
`T` [reduce_sum](#) (`T` const &`tsor`)
- `template<Tensor T>`
`T` [reduce_mean](#) (`T` const &`tsor`)
- `template<Tensor T>`
`T` [clip](#) (`T` &`tsor`, `typename T::value_type` `lower=0`, `typename T::value_type` `upper=1`)
- `template<Tensor T>`
`T` [squeeze](#) (`T` const &`tsor`)
- `template<typename T , typename A = default_allocator<T>>`
`tensor< T, A >` [randn](#) (`std::vector< unsigned long >` const &`shape`, `T` `mean=T{0}`, `T` `stddev=T{1}`)
- `template<typename T , typename A = default_allocator<T>>`
`tensor< T, A >` [truncated_normal](#) (`std::vector< unsigned long >` const &`shape`, `T` `mean=T{0}`, `T` `stddev=T{1}`, `T` `lower=T{0}`, `T` `upper=T{1}`)
- `template<typename T , typename A = default_allocator<T>>`
`tensor< T, A >` [random](#) (`std::vector< unsigned long >` const &`shape`, `T` `min=T{0}`, `T` `max=T{1}`)
- `template<Tensor T>`
`T` [random_like](#) (`T` const &`tsor`, `typename T::value_type` `min=0`, `typename T::value_type` `max=1`)
- `template<typename T , typename A = default_allocator<T>>`
`tensor< T, A >` [glorot_uniform](#) (`std::initializer_list< unsigned long >` `shape`)
- `template<Tensor T>`
`T` [deep_copy](#) (`T` const &`tsor`)
- `template<Tensor T>`
`T` [copy](#) (`T` const &`tsor`)
- `template<Tensor T>`
`T` [concatenate](#) (`T` const &`lhs`, `T` const &`rhs`, `unsigned long` `axis=0`) noexcept
- `template<Tensor T>`
`T` [repmat](#) (`T` const &`tsor`, `unsigned long` `row_rep`, `unsigned long` `col_rep`)
- `template<Tensor T>`
`constexpr bool` [empty](#) (`T` const &`tsor`) noexcept

- `template<typename T , typename A = default_allocator<T>>`
`constexpr tensor< T, A > zeros (std::vector< unsigned long > const &shape)`
- `template<Tensor Tsor>`
`constexpr Tsor zeros_like (Tsor const &tsor)`
- `template<typename T , typename A = default_allocator<T>>`
`constexpr tensor< T, A > ones (std::vector< unsigned long > const &shape)`
- `template<Tensor Tsor>`
`constexpr Tsor ones_like (Tsor const &tsor)`
- `template<Tensor Tsor>`
`auto max (Tsor const &tsor)`
- `template<Tensor Tsor>`
`auto amax (Tsor const &tsor)`
- `template<Tensor Tsor>`
`auto min (Tsor const &tsor)`
- `template<Tensor Tsor>`
`auto amin (Tsor const &tsor)`
- `template<Tensor Tsor>`
`auto sum (Tsor const &tsor)`
- `template<Tensor Tsor>`
`auto mean (Tsor const &tsor)`
- `template<Tensor Tsor>`
`auto norm (Tsor const &tsor)`
- `template<Tensor Tsor>`
`Tsor abs (Tsor const &tsor)`
- `template<Tensor Tsor>`
`Tsor softmax (Tsor const &tsor)`
- `template<Tensor Tsor>`
`bool has_nan (Tsor const &tsor)`
- `template<Tensor Tsor>`
`bool has_inf (Tsor const &tsor)`
- `template<Tensor Tsor>`
`bool is_valid (Tsor const &tsor)`
- `template<Tensor Tsor, typename Function >`
`Tsor reduce (Tsor const &ts, unsigned long axis, typename Tsor::value_type const &init, Function const &func, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`Tsor sum (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`requires std::floating_point< typename Tsor::value_type > Tsor mean (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`requires std::floating_point< typename Tsor::value_type > Tsor variance (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`requires std::floating_point< typename Tsor::value_type > Tsor standard_deviation (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`Tsor max (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`Tsor min (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<typename T , typename A = default_allocator<T>>`
`requires std::floating_point< T > tensor< T, A > linspace (T start, T stop, unsigned long num, bool end-point=true) noexcept`
- `template<class _Tp , class _CharT , class _Traits , class _Alloc >`
`std::basic_istream< _CharT, _Traits > & read_tensor (std::basic_istream< _CharT, _Traits > &__is, tensor< _Tp, _Alloc > &__x)`

- `template<class _Tp, class _CharT, class _Traits, class _Alloc >`
`std::basic_ostream< _CharT, _Traits > & write_tensor (std::basic_ostream< _CharT, _Traits > & __os,
tensor< _Tp, _Alloc > const & __x\)`
- `template<typename T, typename A = default_allocator<T>>`
[tensor< T, A > \[load_tensor\]\(#\) \(std::string const &file_name\)](#)
- `template<Tensor Tsor>`
`void save_tensor (std::string const &file_name, Tsor const &tsor)`
- `template<Variable Var>`
`bool operator== (Var const &lhs, Var const &rhs) noexcept`

Variables

- `constexpr unsigned long version = 20210418UL`
- `constexpr unsigned long __version__ = version`
- `constexpr unsigned long is_windows_platform = 0`
- `constexpr unsigned long debug_mode = 1`
- `constexpr unsigned long blas_mode = 0`
- `constexpr unsigned long cuda_mode = 0`
- `int visible_device = 0`
- `unsigned long cuda_gemm_threshold = 0UL`
- `constexpr double eps = 1.0e-8`
- `int learning_phase = 1`
- `template<class T >`
`constexpr bool is_constant_v = is_constant<T>::value`
- `template<typename T >`
`concept Constant = is_constant_v<T>`
- `static constexpr auto make_unary_operator`
- `static constexpr auto make_binary_operator`
- `template<class T >`
`constexpr bool is_operator_v = is_operator<T>::value`
- `template<typename T >`
`concept Operator = is_operator_v<T>`
- `template<typename T >`
`concept Expression = Operator<T> || Variable<T> || Place_Holder<T> || Constant<T> || Value<T>`
- `template<class T >`
`constexpr bool is_place_holder_v = is_place_holder<T>::value`
- `template<typename T >`
`concept Place_Holder = is_place_holder_v<T>`
- `auto lstm`
- `static unsigned long random_seed = std::chrono::system_clock::now().time_since_epoch().count()`
- `static std::mt19937 random_generator {random_seed}`
- `template<class T >`
`constexpr bool is_tensor_v = is_tensor<T>::value`
- `template<typename T >`
`concept Tensor = is_tensor_v<T>`
- `template<class T >`
`constexpr bool is_value_v = is_value<T>::value`
- `template<typename T >`
`concept Value = is_value_v<T>`
- `template<class T >`
`constexpr bool is_variable_v = is_variable<T>::value`
- `template<typename T >`
`concept Variable = is_variable_v<T>`

5.1.1 Typedef Documentation

5.1.1.1 ada_delta

```
template<typename Loss , typename T >  
using ceras::ada_delta = typedef adadelta< Loss, T >
```

5.1.1.2 ada_grad

```
template<typename Loss , typename T >  
using ceras::ada_grad = typedef adagrad<Loss, T>
```

5.1.1.3 default_allocator

```
template<typename T >  
using ceras::default_allocator = typedef std::allocator<T>
```

5.1.1.4 rms_prop

```
template<typename Loss , typename T >  
using ceras::rms_prop = typedef rmsprop< Loss, T >
```

5.1.2 Function Documentation

5.1.2.1 abs() [1/2]

```
template<Expression Ex>  
constexpr auto ceras::abs (  
    Ex const & ex ) [constexpr], [noexcept]
```


5.1.2.2 abs() [2/2]

```
template<Tensor Tsor>
Tsor ceras::abs (
    Tsor const & tsor )
```

5.1.2.3 abs_loss()

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::abs_loss (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.4 add()

```
template<Tensor Tsor>
Tsor ceras::add (
    Tsor const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.5 amax()

```
template<Tensor Tsor>
auto ceras::amax (
    Tsor const & tsor )
```

5.1.2.6 amin()

```
template<Tensor Tsor>
auto ceras::amin (
    Tsor const & tsor )
```

5.1.2.7 as_tensor()

```
template<typename T , typename A = default_allocator<T>>
constexpr tensor<T, A> ceras::as_tensor (
    T val ) [constexpr], [noexcept]
```

5.1.2.8 average_pooling_2d()

```
auto ceras::average_pooling_2d (
    unsigned long stride ) [inline], [noexcept]
```

5.1.2.9 batch_normalization()

```
template<typename T >
requires std::floating_point<T> auto ceras::batch_normalization (
    T const momentum = 0.98 ) [inline], [noexcept]
```

5.1.2.10 BatchNormalization()

```
auto ceras::BatchNormalization (
    std::vector< unsigned long > const & shape,
    float threshold = 0.95f ) [inline]
```

5.1.2.11 clip() [1/2]

```
template<typename Float >
requires std::floating_point<Float> constexpr auto ceras::clip (
    Float lower,
    Float upper = std::numeric_limits<Float>::max() ) [constexpr], [noexcept]
```

5.1.2.12 clip() [2/2]

```
template<Tensor Tsor>
Tsor ceras::clip (
    Tsor & tsor,
    typename Tsor::value_type lower = 0,
    typename Tsor::value_type upper = 1 )
```

5.1.2.13 concat() [1/2]

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::concat (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.14 concat() [2/2]

```
auto ceras::concat (
    unsigned long axe = -1 ) [inline]
```

5.1.2.15 concatenate() [1/3]

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::concatenate (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.16 concatenate() [2/3]

```
template<Tensor Tsor>
Tsor ceras::concatenate (
    Tsor const & lhs,
    Tsor const & rhs,
    unsigned long axis = 0 ) [noexcept]
```

5.1.2.17 concatenate() [3/3]

```
auto ceras::concatenate (
    unsigned long axe = -1 ) [inline]
```

5.1.2.18 Conv2D()

```
auto ceras::Conv2D (
    unsigned long output_channels,
    std::vector< unsigned long > const & kernel_size,
    std::vector< unsigned long > const & input_shape,
    std::string const & padding = "valid",
    std::vector< unsigned long > const & strides = {1,1} ) [inline]
```

5.1.2.19 conv2d()

```

auto ceras::conv2d (
    unsigned long row_input,
    unsigned long col_input,
    unsigned long const row_stride = 1,
    unsigned long const col_stride = 1,
    unsigned long const row_dilation = 1,
    unsigned long const col_dilation = 1,
    std::string const & padding = "valid" ) [inline], [noexcept]

```

5.1.2.20 copy() [1/2]

```

template<Expression Lhs_Expression, Variable Rhs_Variable>
constexpr auto ceras::copy (
    Lhs_Expression const & lhs_ex,
    Rhs_Variable const & rhs_va ) [constexpr], [noexcept]

```

5.1.2.21 copy() [2/2]

```

template<Tensor Tsor>
Tsor ceras::copy (
    Tsor const & tsor )

```

5.1.2.22 cross_entropy()

```

template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::cross_entropy (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]

```

5.1.2.23 cross_entropy_loss()

```

template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::cross_entropy_loss (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]

```

5.1.2.24 deep_copy()

```
template<Tensor Tsor>
Tsor ceras::deep_copy (
    Tsor const & tsor )
```

5.1.2.25 Dense()

```
auto ceras::Dense (
    unsigned long output_size,
    unsigned long input_size ) [inline]
```

5.1.2.26 drop_out()

```
template<typename T >
requires std::floating_point<T> auto ceras::drop_out (
    T const factor ) [inline], [noexcept]
```

5.1.2.27 elementwise_divide()

```
template<Tensor Tsor>
Tsor ceras::elementwise_divide (
    Tsor const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.28 elementwise_product() [1/2]

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::elementwise_product (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.29 elementwise_product() [2/2]

```
template<Tensor Tsor>
Tsor ceras::elementwise_product (
    Tsor const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.30 elu()

```
template<typename T >
requires std::floating_point<T> auto ceras::elu (
    T const alpha ) [noexcept]
```

5.1.2.31 empty()

```
template<Tensor Tsor>
constexpr bool ceras::empty (
    Tsor const & tsor ) [constexpr], [noexcept]
```

5.1.2.32 exp()

```
template<Expression Ex>
constexpr auto ceras::exp (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.33 exponential()

```
template<Expression Ex>
auto ceras::exponential (
    Ex const & ex ) [inline], [noexcept]
```

5.1.2.34 flatten()

```
template<Expression Ex>
constexpr auto ceras::flatten (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.35 gelu()

```
template<Expression Ex>
auto ceras::gelu (
    Ex const & ex ) [inline], [noexcept]
```

5.1.2.36 `gemm()` [1/2]

```
template<typename T >
requires std::floating_point<T> void ceras::gemm (
    T const * A,
    bool a_transposed,
    T const * B,
    bool b_transposed,
    unsigned long m,
    unsigned long n,
    unsigned long k,
    T * C )
```

5.1.2.37 `gemm()` [2/2]

```
template<typename T >
requires std::floating_point<T> void ceras::gemm (
    view_2d< T > const & x,
    view_2d< T > const & y,
    view_2d< T > & ans )
```

5.1.2.38 `gemm_cpu()`

```
template<typename T >
requires std::floating_point<T> void ceras::gemm_cpu (
    T const * A,
    bool a_transposed,
    T const * B,
    bool b_transposed,
    unsigned long m,
    unsigned long n,
    unsigned long k,
    T * C )
```

5.1.2.39 `get_default_session()`

```
template<Tensor Tsor>
std::reference_wrapper< session< Tsor > > ceras::get_default_session ( )
```

5.1.2.40 `glorot_uniform()`

```
template<typename T , typename A = default_allocator<T>>
tensor<T,A> ceras::glorot_uniform (
    std::initializer_list< unsigned long > shape )
```

5.1.2.41 hadamard_product() [1/2]

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::hadamard_product (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.42 hadamard_product() [2/2]

```
template<Tensor Tsor>
Tsor ceras::hadamard_product (
    Tsor const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.43 hard_sigmoid()

```
template<Expression Ex>
auto ceras::hard_sigmoid (
    Ex const & ex ) [inline], [noexcept]
```

5.1.2.44 has_inf()

```
template<Tensor Tsor>
bool ceras::has_inf (
    Tsor const & tsor )
```

5.1.2.45 has_nan()

```
template<Tensor Tsor>
bool ceras::has_nan (
    Tsor const & tsor )
```

5.1.2.46 hinge_loss()

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::hinge_loss (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```


5.1.2.47 identity()

```
template<Expression Ex>
constexpr auto ceras::identity (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.48 img2col()

```
auto ceras::img2col (
    unsigned long const row_kernel,
    unsigned long col_kernel = -1,
    unsigned long const row_padding = 0,
    unsigned long col_padding = 0,
    unsigned long const row_stride = 1,
    unsigned long const col_stride = 1,
    unsigned long const row_dilation = 1,
    unsigned long const col_dilation = 1 ) [inline], [noexcept]
```

5.1.2.49 Input()

```
auto ceras::Input ( ) [inline]
```

5.1.2.50 instance_normalization()

```
template<typename T >
requires std::floating_point<T> auto ceras::instance_normalization (
    T const momentum = 0.98 ) [inline], [noexcept]
```

5.1.2.51 is_valid()

```
template<Tensor Tsor>
bool ceras::is_valid (
    Tsor const & tsor )
```

5.1.2.52 leaky_relu()

```
template<typename T >
requires std::floating_point<T> auto ceras::leaky_relu (
    T const factor ) [noexcept]
```

5.1.2.53 linspace()

```
template<typename T , typename A = default_allocator<T>>
requires std::floating_point<T> tensor<T,A> ceras::linspace (
    T start,
    T stop,
    unsigned long num,
    bool endpoint = true ) [noexcept]
```

5.1.2.54 load_tensor()

```
template<typename T , typename A = default_allocator<T>>
tensor<T,A> ceras::load_tensor (
    std::string const & file_name )
```

5.1.2.55 log()

```
template<Expression Ex>
constexpr auto ceras::log (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.56 mae()

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::mae (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.57 max() [1/2]

```
template<Tensor Tsor>
Tsor ceras::max (
    Tsor const & ts,
    unsigned long axis,
    bool keepdims = false ) [noexcept]
```

5.1.2.58 max() [2/2]

```
template<Tensor Tsor>
auto ceras::max (
    Tsor const & tsor )
```

5.1.2.59 max_pooling_2d()

```
auto ceras::max_pooling_2d (
    unsigned long stride ) [inline], [noexcept]
```

5.1.2.60 maximum()

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::maximum (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.61 mean() [1/2]

```
template<Tensor Tsor>
requires std::floating_point<typename Tsor::value_type> Tsor ceras::mean (
    Tsor const & ts,
    unsigned long axis,
    bool keepdims = false ) [noexcept]
```

5.1.2.62 mean() [2/2]

```
template<Tensor Tsor>
auto ceras::mean (
    Tsor const & tsor )
```

5.1.2.63 mean_absolute_error()

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::mean_absolute_error (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.64 mean_reduce()

```
template<Expression Ex>
constexpr auto ceras::mean_reduce (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.65 mean_squared_error()

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::mean_squared_error (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.66 mean_squared_logarithmic_error()

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::mean_squared_logarithmic_error (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.67 min() [1/2]

```
template<Tensor Tsor>
Tsor ceras::min (
    Tsor const & ts,
    unsigned long axis,
    bool keepdims = false ) [noexcept]
```

5.1.2.68 min() [2/2]

```
template<Tensor Tsor>
auto ceras::min (
    Tsor const & tsor )
```

5.1.2.69 minus() [1/2]

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::minus (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.70 minus() [2/2]

```
template<Tensor Tsor>
Tsor ceras::minus (
    Tsor const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.71 mse()

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::mse (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.72 multiply() [1/2]

```
template<Tensor Tsor>
Tsor ceras::multiply (
    Tsor const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.73 multiply() [2/2]

```
template<Tensor Tsor>
void ceras::multiply (
    Tsor const & lhs,
    Tsor const & rhs,
    Tsor & ans ) [noexcept]
```

5.1.2.74 negative()

```
template<Expression Ex>
constexpr auto ceras::negative (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.75 norm()

```
template<Tensor Tsor>
auto ceras::norm (
    Tsor const & tsor )
```

5.1.2.76 normalization_batch()

```
template<typename T = double>
requires std::floating_point<T> auto ceras::normalization_batch (
    T const momentum = 0.98 ) [inline], [noexcept]
```

5.1.2.77 normalization_instance()

```
template<typename T = double>
requires std::floating_point<T> auto ceras::normalization_instance (
    T const momentum = 0.98 ) [inline], [noexcept]
```

5.1.2.78 ones()

```
template<typename T , typename A = default_allocator<T>>
constexpr tensor<T,A> ceras::ones (
    std::vector< unsigned long > const & shape ) [constexpr]
```

5.1.2.79 ones_like()

```
template<Tensor Tsor>
constexpr Tsor ceras::ones_like (
    Tsor const & tsor ) [constexpr]
```

5.1.2.80 operator*() [1/4]

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
auto ceras::operator* (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [noexcept]
```

5.1.2.81 operator*() [2/4]

```
template<Tensor Tsor>
Tsor ceras::operator* (
    Tsor const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.82 operator*() [3/4]

```
template<Tensor Tsor>
Tsor ceras::operator* (
    Tsor const & lhs,
    typename Tsor::value_type const & rhs ) [noexcept]
```

5.1.2.83 operator*() [4/4]

```
template<Tensor Tsor>
Tsor ceras::operator* (
    typename Tsor::value_type const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.84 operator+() [1/4]

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::operator+ (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.85 operator+() [2/4]

```
template<Tensor Tsor>
Tsor ceras::operator+ (
    Tsor const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.86 operator+() [3/4]

```
template<Tensor Tsor>
Tsor ceras::operator+ (
    Tsor const & lhs,
    typename Tsor::value_type const & rhs ) [noexcept]
```

5.1.2.87 operator+() [4/4]

```
template<Tensor Tsor>
Tsor ceras::operator+ (
    typename Tsor::value_type const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.88 operator-() [1/4]

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::operator- (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.89 operator-() [2/4]

```
template<Tensor Tsor>
Tsor ceras::operator- (
    Tsor const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.90 operator-() [3/4]

```
template<Tensor Tsor>
Tsor ceras::operator- (
    Tsor const & lhs,
    typename Tsor::value_type const & rhs ) [noexcept]
```

5.1.2.91 operator-() [4/4]

```
template<Tensor Tsor>
Tsor ceras::operator- (
    typename Tsor::value_type const & lhs,
    Tsor const & rhs ) [noexcept]
```

5.1.2.92 operator/()

```
template<Tensor Tsor>
Tsor ceras::operator/ (
    Tsor const & lhs,
    typename Tsor::value_type const & rhs ) [noexcept]
```

5.1.2.93 operator<<()

```
template<Tensor Tsor, typename CharT , typename Traits >
std::basic_ostream<CharT, Traits>& ceras::operator<< (
    std::basic_ostream< CharT, Traits > & os_,
    Tsor const & tsor )
```


5.1.2.94 operator==()

```
template<Variable Var>
bool ceras::operator==(
    Var const & lhs,
    Var const & rhs ) [noexcept]
```

5.1.2.95 plus()

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::plus (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.96 randn()

```
template<typename T , typename A = default_allocator<T>>
tensor<T,A> ceras::randn (
    std::vector< unsigned long > const & shape,
    T mean = T{0},
    T stddev = T{1} )
```

5.1.2.97 random()

```
template<typename T , typename A = default_allocator<T>>
tensor<T,A> ceras::random (
    std::vector< unsigned long > const & shape,
    T min = T{0},
    T max = T{1} )
```

5.1.2.98 random_like()

```
template<Tensor Tsor>
Tsor ceras::random_like (
    Tsor const & tsor,
    typename Tsor::value_type min = 0,
    typename Tsor::value_type max = 1 )
```

5.1.2.99 read_tensor()

```
template<class _Tp , class _CharT , class _Traits , class _Alloc >
std::basic_istream<_CharT, _Traits>& ceras::read_tensor (
    std::basic_istream< _CharT, _Traits > & __is,
    tensor< _Tp, _Alloc > & __x )
```

5.1.2.100 reduce()

```
template<Tensor Tsor, typename Function >
Tsor ceras::reduce (
    Tsor const & ts,
    unsigned long axis,
    typename Tsor::value_type const & init,
    Function const & func,
    bool keepdims = false ) [noexcept]
```

5.1.2.101 reduce_mean() [1/2]

```
template<Expression Ex>
constexpr auto ceras::reduce_mean (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.102 reduce_mean() [2/2]

```
template<Tensor Tsor>
Tsor ceras::reduce_mean (
    Tsor const & tsor )
```

5.1.2.103 reduce_sum() [1/2]

```
template<Expression Ex>
constexpr auto ceras::reduce_sum (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.104 reduce_sum() [2/2]

```
template<Tensor Tsor>
Tsor ceras::reduce_sum (
    Tsor const & tsor )
```

5.1.2.105 relu()

```
template<Expression Ex>
auto ceras::relu (
    Ex const & ex ) [noexcept]
```

5.1.2.106 repeat()

```
template<Tensor Tsor>
Tsor ceras::repeat (
    Tsor const & tsor,
    unsigned long n )
```

5.1.2.107 repmat()

```
template<Tensor Tsor>
Tsor ceras::repmat (
    Tsor const & tsor,
    unsigned long row_rep,
    unsigned long col_rep )
```

5.1.2.108 reshape() [1/2]

```
auto ceras::reshape (
    std::vector< unsigned long > const & new_shape,
    bool include_batch_flag = true ) [inline], [noexcept]
```

5.1.2.109 reshape() [2/2]

```
template<Tensor Tsor>
Tsor ceras::reshape (
    Tsor const & ts,
    std::vector< unsigned long > const & new_shape )
```

5.1.2.110 save_tensor()

```
template<Tensor Tsor>
void ceras::save_tensor (
    std::string const & file_name,
    Tsor const & tsor )
```

5.1.2.111 `selu()`

```
template<Expression Ex>
auto ceras::selu (
    Ex const & ex ) [inline], [noexcept]
```

5.1.2.112 `sigmoid()`

```
template<Expression Ex>
auto ceras::sigmoid (
    Ex const & ex ) [inline], [noexcept]
```

5.1.2.113 `softmax()` [1/2]

```
template<Expression Ex>
constexpr auto ceras::softmax (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.114 `softmax()` [2/2]

```
template<Tensor Tsor>
Tsor ceras::softmax (
    Tsor const & tsor )
```

5.1.2.115 `softplus()`

```
template<Expression Ex>
auto ceras::softplus (
    Ex const & ex ) [inline], [noexcept]
```

5.1.2.116 `softsign()`

```
template<Expression Ex>
auto ceras::softsign (
    Ex const & ex ) [inline], [noexcept]
```

5.1.2.117 square()

```
template<Expression Ex>
constexpr auto ceras::square (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.118 squared_loss()

```
template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto ceras::squared_loss (
    Lhs_Expression const & lhs_ex,
    Rhs_Expression const & rhs_ex ) [constexpr], [noexcept]
```

5.1.2.119 squeeze()

```
template<Tensor Tsor>
Tsor ceras::squeeze (
    Tsor const & tsor )
```

5.1.2.120 standard_deviation()

```
template<Tensor Tsor>
requires std::floating_point<typename Tsor::value_type> Tsor ceras::standard_deviation (
    Tsor const & ts,
    unsigned long axis,
    bool keepdims = false ) [noexcept]
```

5.1.2.121 sum() [1/2]

```
template<Tensor Tsor>
Tsor ceras::sum (
    Tsor const & ts,
    unsigned long axis,
    bool keepdims = false ) [noexcept]
```

5.1.2.122 sum() [2/2]

```
template<Tensor Tsor>
auto ceras::sum (
    Tsor const & tsor )
```

5.1.2.123 sum_reduce()

```
template<Expression Ex>
constexpr auto ceras::sum_reduce (
    Ex const & ex ) [constexpr], [noexcept]
```

5.1.2.124 tanh()

```
template<Expression Ex>
auto ceras::tanh (
    Ex const & ex ) [inline], [noexcept]
```

5.1.2.125 transpose()

```
template<Expression Ex>
auto ceras::transpose (
    Ex const & ex ) [noexcept]
```

5.1.2.126 truncated_normal()

```
template<typename T , typename A = default_allocator<T>>
tensor<T,A> ceras::truncated_normal (
    std::vector< unsigned long > const & shape,
    T mean = T{0},
    T stddev = T{1},
    T lower = T{0},
    T upper = T{1} )
```

5.1.2.127 up_sampling_2d()

```
auto ceras::up_sampling_2d (
    unsigned long stride ) [inline], [noexcept]
```

5.1.2.128 update_cuda_gemm_threshold()

```
void ceras::update_cuda_gemm_threshold ( ) [inline]
```

5.1.2.129 variance()

```
template<Tensor Tsor>
requires std::floating_point<typename Tsor::value_type> Tsor ceras::variance (
    Tsor const & ts,
    unsigned long axis,
    bool keepdims = false ) [noexcept]
```

5.1.2.130 write_tensor()

```
template<class _Tp , class _CharT , class _Traits , class _Alloc >
std::basic_ostream<_CharT, _Traits>& ceras::write_tensor (
    std::basic_ostream< _CharT, _Traits > & __os,
    tensor< _Tp, _Alloc > const & __x )
```

5.1.2.131 zeros()

```
template<typename T , typename A = default_allocator<T>>
constexpr tensor<T,A> ceras::zeros (
    std::vector< unsigned long > const & shape ) [constexpr]
```

5.1.2.132 zeros_like()

```
template<Tensor Tsor>
constexpr Tsor ceras::zeros_like (
    Tsor const & tsor ) [constexpr]
```

5.1.3 Variable Documentation**5.1.3.1 __version__**

```
constexpr unsigned long ceras::__version__ = version [inline], [constexpr]
```

5.1.3.2 blas_mode

```
constexpr unsigned long ceras::blas_mode = 0 [inline], [constexpr]
```

5.1.3.3 Constant

```
template<typename T >  
concept ceras::Constant = is\_constant\_v<T>
```

5.1.3.4 cuda_gemm_threshold

```
unsigned long ceras::cuda_gemm_threshold = 0UL [inline]
```

5.1.3.5 cuda_mode

```
constexpr unsigned long ceras::cuda_mode = 0 [inline], [constexpr]
```

5.1.3.6 debug_mode

```
constexpr unsigned long ceras::debug_mode = 1 [inline], [constexpr]
```

5.1.3.7 eps

```
constexpr double ceras::eps = 1.0e-8 [inline], [constexpr]
```

5.1.3.8 Expression

```
template<typename T >  
concept ceras::Expression = Operator<T> || Variable<T> || Place\_Holder<T> || Constant<T> ||  
Value<T>
```

5.1.3.9 is_constant_v

```
template<class T >  
constexpr bool ceras::is_constant_v = is\_constant<T>::value [inline], [constexpr]
```


5.1.3.10 is_operator_v

```
template<class T >
constexpr bool ceras::is_operator_v = is_operator<T>::value [inline], [constexpr]
```

5.1.3.11 is_place_holder_v

```
template<class T >
constexpr bool ceras::is_place_holder_v = is_place_holder<T>::value [inline], [constexpr]
```

5.1.3.12 is_tensor_v

```
template<class T >
constexpr bool ceras::is_tensor_v = is_tensor<T>::value [inline], [constexpr]
```

5.1.3.13 is_value_v

```
template<class T >
constexpr bool ceras::is_value_v = is_value<T>::value [inline], [constexpr]
```

5.1.3.14 is_variable_v

```
template<class T >
constexpr bool ceras::is_variable_v = is_variable<T>::value [inline], [constexpr]
```

5.1.3.15 is_windows_platform

```
constexpr unsigned long ceras::is_windows_platform = 0 [inline], [constexpr]
```

Parameters

<i>is_windows_platform</i>	A constexpr helping ceras to select different behaviours. 1 for windows platform and 0 for non-windows platform.
----------------------------	------------------------------------------------------------------------------------------------------------------

Example usage:

```
if constexpr( is_windows_platform )
{
```

```

        call_windows_method();
    }
    else
    {
        call_linux_method();
    }

```

5.1.3.16 learning_phase

```
int ceras::learning_phase = 1 [inline]
```

5.1.3.17 lstm

```
auto ceras::lstm [inline]
```

5.1.3.18 make_binary_operator

```
constexpr auto ceras::make_binary_operator [static], [constexpr]
```

Initial value:

```

= [] ( auto const& binary_forward_action, auto const& binary_backward_action, std::string const&
    name="Anonymous Binary Operator", std::function<void()> const& reset_action=[](){} ) noexcept
{
    return [&binary_forward_action, &binary_backward_action, &name, &reset_action] ( auto const& lhs_op,
    auto const& rhs_op ) noexcept
    {
        auto ans = binary_operator( lhs_op, rhs_op, binary_forward_action, binary_backward_action,
        reset_action );
        ans.name_ = name;
        return ans;
    };
}

```

5.1.3.19 make_unary_operator

```
constexpr auto ceras::make_unary_operator [static], [constexpr]
```

Initial value:

```

= [] ( auto const& unary_forward_action, auto const& unary_backward_action, std::string const&
    name="Anonymous Unary Operator", std::function<void()> reset_action = [](){} ) noexcept
{
    return [&unary_forward_action, &unary_backward_action, &name, &reset_action] ( auto const& op )
    noexcept
    {
        auto ans = unary_operator( op, unary_forward_action, unary_backward_action, reset_action );
        ans.name_ = name;
        return ans;
    };
}

```

5.1.3.20 Operator

```
template<typename T >  
concept ceras::Operator = is_operator_v<T>
```

5.1.3.21 Place_Holder

```
template<typename T >  
concept ceras::Place_Holder = is_place_holder_v<T>
```

5.1.3.22 random_generator

```
std::mt19937 ceras::random_generator {random_seed} [static]
```

5.1.3.23 random_seed

```
unsigned long ceras::random_seed = std::chrono::system_clock::now().time_since_epoch().count()  
[static]
```

5.1.3.24 Tensor

```
template<typename T >  
concept ceras::Tensor = is_tensor_v<T>
```

5.1.3.25 Value

```
template<typename T >  
concept ceras::Value = is_value_v<T>
```

5.1.3.26 Variable

```
template<typename T >  
concept ceras::Variable = is_variable_v<T>
```

5.1.3.27 version

```
constexpr unsigned long ceras::version = 20210418UL [inline], [constexpr]
```

5.1.3.28 visible_device

```
int ceras::visible_device = 0 [inline]
```

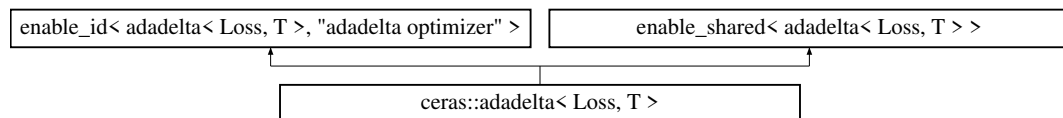

Chapter 6

Class Documentation

6.1 ceras::adadelta< Loss, T > Struct Template Reference

```
#include <optimizer.hpp>
```

Inheritance diagram for ceras::adadelta< Loss, T >:



Public Types

- typedef [tensor](#)< T > [tensor_type](#)

Public Member Functions

- [adadelta](#) (Loss &loss, std::size_t batch_size, T rho=0.9) noexcept
- void [forward](#) ()

Public Attributes

- Loss & [loss_](#)
- T [rho_](#)
- T [learning_rate_](#)
- unsigned long [iterations_](#)

6.1.1 Member Typedef Documentation

6.1.1.1 tensor_type

```
template<typename Loss , typename T >
typedef tensor< T > ceras::adadelta< Loss, T >::tensor_type
```

6.1.2 Constructor & Destructor Documentation

6.1.2.1 adadelta()

```
template<typename Loss , typename T >
ceras::adadelta< Loss, T >::adadelta (
    Loss & loss,
    std::size_t batch_size,
    T rho = 0.9 ) [inline], [noexcept]
```

6.1.3 Member Function Documentation

6.1.3.1 forward()

```
template<typename Loss , typename T >
void ceras::adadelta< Loss, T >::forward ( ) [inline]
```

6.1.4 Member Data Documentation

6.1.4.1 iterations_

```
template<typename Loss , typename T >
unsigned long ceras::adadelta< Loss, T >::iterations_
```

6.1.4.2 learning_rate_

```
template<typename Loss , typename T >
T ceras::adadelta< Loss, T >::learning_rate_
```

6.1.4.3 loss_

```
template<typename Loss , typename T >
Loss& ceras::adadelat< Loss, T >::loss_
```

6.1.4.4 rho_

```
template<typename Loss , typename T >
T ceras::adadelat< Loss, T >::rho_
```

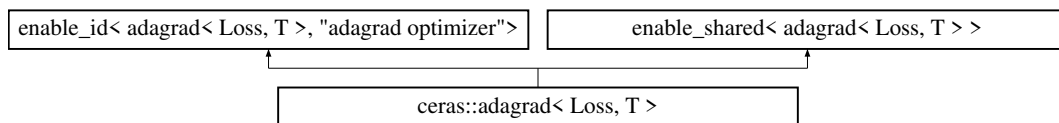
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/optimizer.hpp

6.2 ceras::adagrad< Loss, T > Struct Template Reference

```
#include <optimizer.hpp>
```

Inheritance diagram for ceras::adagrad< Loss, T >:



Public Types

- typedef `tensor< T >` `tensor_type`

Public Member Functions

- `adagrad` (Loss &loss, std::size_t batch_size, T learning_rate=1.0e-1, T decay=0.0) noexcept
- void `forward` ()

Public Attributes

- Loss & `loss_`
- T `learning_rate_`
- T `decay_`
- unsigned long `iterations_`

6.2.1 Member Typedef Documentation

6.2.1.1 tensor_type

```
template<typename Loss , typename T >
typedef tensor< T > ceras::adagrad< Loss, T >::tensor\_type
```

6.2.2 Constructor & Destructor Documentation

6.2.2.1 adagrad()

```
template<typename Loss , typename T >
ceras::adagrad< Loss, T >::adagrad (
    Loss & loss,
    std::size_t batch_size,
    T learning_rate = 1.0e-1,
    T decay = 0.0 ) [inline], [noexcept]
```

6.2.3 Member Function Documentation

6.2.3.1 forward()

```
template<typename Loss , typename T >
void ceras::adagrad< Loss, T >::forward ( ) [inline]
```

6.2.4 Member Data Documentation

6.2.4.1 decay_

```
template<typename Loss , typename T >
T ceras::adagrad< Loss, T >::decay_
```

6.2.4.2 iterations_

```
template<typename Loss , typename T >
unsigned long ceras::adagrad< Loss, T >::iterations_
```


6.2.4.3 learning_rate_

```
template<typename Loss , typename T >
T ceras::adagrad< Loss, T >::learning_rate_
```

6.2.4.4 loss_

```
template<typename Loss , typename T >
Loss& ceras::adagrad< Loss, T >::loss_
```

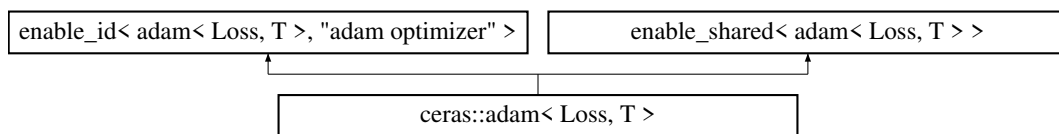
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/optimizer.hpp

6.3 ceras::adam< Loss, T > Struct Template Reference

```
#include <optimizer.hpp>
```

Inheritance diagram for ceras::adam< Loss, T >:



Public Types

- typedef [tensor](#)< T > [tensor_type](#)

Public Member Functions

- [adam](#) (Loss &loss, std::size_t batch_size, T learning_rate=1.0e-1, T beta_1=0.9, T beta_2=0.999, bool amsgrad=false) noexcept
- void [forward](#) ()

Public Attributes

- Loss & [loss_](#)
- T [learning_rate_](#)
- T [beta_1_](#)
- T [beta_2_](#)
- bool [amsgrad_](#)
- unsigned long [iterations_](#)

6.3.1 Member Typedef Documentation

6.3.1.1 tensor_type

```
template<typename Loss , typename T >
typedef tensor< T > ceras::adam< Loss, T >::tensor_type
```

6.3.2 Constructor & Destructor Documentation

6.3.2.1 adam()

```
template<typename Loss , typename T >
ceras::adam< Loss, T >::adam (
    Loss & loss,
    std::size_t batch_size,
    T learning_rate = 1.0e-1,
    T beta_1 = 0.9,
    T beta_2 = 0.999,
    bool amsgrad = false ) [inline], [noexcept]
```

6.3.3 Member Function Documentation

6.3.3.1 forward()

```
template<typename Loss , typename T >
void ceras::adam< Loss, T >::forward ( ) [inline]
```

6.3.4 Member Data Documentation

6.3.4.1 amsgrad_

```
template<typename Loss , typename T >
bool ceras::adam< Loss, T >::amsgrad_
```

6.3.4.2 beta_1_

```
template<typename Loss , typename T >
T ceras::adam< Loss, T >::beta_1_
```

6.3.4.3 beta_2_

```
template<typename Loss , typename T >
T ceras::adam< Loss, T >::beta_2_
```

6.3.4.4 iterations_

```
template<typename Loss , typename T >
unsigned long ceras::adam< Loss, T >::iterations_
```

6.3.4.5 learning_rate_

```
template<typename Loss , typename T >
T ceras::adam< Loss, T >::learning_rate_
```

6.3.4.6 loss_

```
template<typename Loss , typename T >
Loss& ceras::adam< Loss, T >::loss_
```

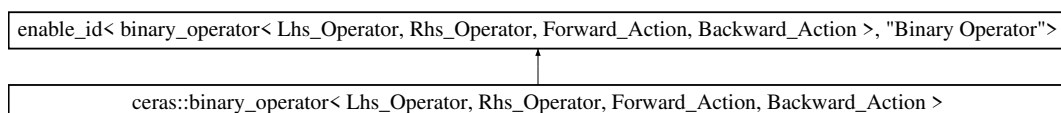
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/optimizer.hpp

6.4 ceras::binary_operator< Lhs_Operator, Rh Operator, Forward_Action, Backward_Action > Struct Template Reference

```
#include <operation.hpp>
```

Inheritance diagram for ceras::binary_operator< Lhs_Operator, Rh Operator, Forward_Action, Backward_Action >:



Public Types

- using [tensor_type](#) = typename [tensor_deduction](#)< Lhs_Operator, Rhs_Operator >::[tensor_type](#)

Public Member Functions

- [binary_operator](#) (Lhs_Operator const &lhs_op, Rhs_Operator const &rhs_op, Forward_Action const &forward_action, Backward_Action const &backward_action, std::function< void()> const &reset_action) noexcept
- auto [forward](#) ()
- void [backward](#) ([tensor_type](#) const &grad)
- void [reset_states](#) ()

Public Attributes

- Lhs_Operator [lhs_op_](#)
- Rhs_Operator [rhs_op_](#)
- Forward_Action [forward_action_](#)
- Backward_Action [backward_action_](#)
- std::function< void()> [reset_action_](#)
- [tensor_type](#) [lhs_input_data_](#)
- [tensor_type](#) [rhs_input_data_](#)
- [tensor_type](#) [output_data_](#)

6.4.1 Member Typedef Documentation

6.4.1.1 [tensor_type](#)

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
using ceras::binary\_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >↔
::tensor\_type = typename tensor\_deduction<Lhs_Operator, Rhs_Operator>::tensor\_type
```

6.4.2 Constructor & Destructor Documentation

6.4.2.1 [binary_operator](#)()

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
ceras::binary\_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >::binary\_operator
(
    Lhs_Operator const & lhs_op,
    Rhs_Operator const & rhs_op,
    Forward_Action const & forward_action,
    Backward_Action const & backward_action,
    std::function< void()> const & reset_action ) [inline], [noexcept]
```

6.4.3 Member Function Documentation

6.4.3.1 backward()

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
void ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >↵
::backward (
    tensor_type const & grad ) [inline]
```

6.4.3.2 forward()

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
auto ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >↵
::forward ( ) [inline]
```

6.4.3.3 reset_states()

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
void ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >↵
::reset_states ( ) [inline]
```

6.4.4 Member Data Documentation

6.4.4.1 backward_action_

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
Backward_Action ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward↵
_Action >::backward_action_
```

6.4.4.2 forward_action_

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
Forward_Action ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward↵
_Action >::forward_action_
```

6.4.4.3 lhs_input_data_

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
tensor_type ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_↵
Action >::lhs_input_data_
```

6.4.4.4 lhs_op_

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
Lhs_Operator ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_↵
Action >::lhs_op_
```

6.4.4.5 output_data_

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
tensor_type ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_↵
Action >::output_data_
```

6.4.4.6 reset_action_

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
std::function<void()> ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action,
Backward_Action >::reset_action_
```

6.4.4.7 rhs_input_data_

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
tensor_type ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_↵
Action >::rhs_input_data_
```

6.4.4.8 rhs_op_

```
template<typename Lhs_Operator , typename Rhs_Operator , typename Forward_Action , typename
Backward_Action >
Rhs_Operator ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_↵
Action >::rhs_op_
```

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

- /data/structured_folders/workspace/github.repo/ceras/include/operation.hpp

6.5 ceras::constant< Tsor > Struct Template Reference

```
#include <constant.hpp>
```

Public Member Functions

- void [backward](#) (auto) const
- Tsor [forward](#) () const
- auto [shape](#) () const

Public Attributes

- Tsor [data_](#)

6.5.1 Member Function Documentation

6.5.1.1 backward()

```
template<Tensor Tsor>
void ceras::constant< Tsor >::backward (
    auto ) const [inline]
```

6.5.1.2 forward()

```
template<Tensor Tsor>
Tsor ceras::constant< Tsor >::forward ( ) const [inline]
```

6.5.1.3 shape()

```
template<Tensor T>
auto ceras::constant< T>::shape ( ) const [inline]
```

6.5.2 Member Data Documentation

6.5.2.1 data_

```
template<Tensor T>
T ceras::constant< T>::data_
```

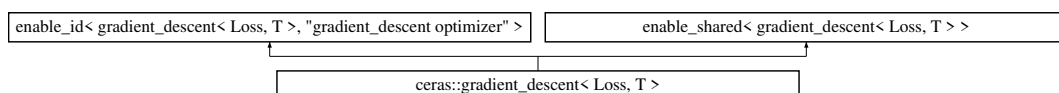
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/[constant.hpp](#)

6.6 ceras::gradient_descent< Loss, T > Struct Template Reference

```
#include <optimizer.hpp>
```

Inheritance diagram for ceras::gradient_descent< Loss, T >:



Public Types

- typedef [tensor](#)< T > [tensor_type](#)

Public Member Functions

- [gradient_descent](#) (Loss &loss, std::size_t batch_size, T learning_rate=1.0e-3, T momentum=0.0) noexcept
- void [forward](#) ()

Public Attributes

- Loss & [loss_](#)
- T [learning_rate_](#)
- T [momentum_](#)

6.6.1 Member Typedef Documentation

6.6.1.1 tensor_type

```
template<typename Loss , typename T >
typedef tensor< T > ceras::gradient\_descent< Loss, T >::tensor\_type
```

6.6.2 Constructor & Destructor Documentation

6.6.2.1 gradient_descent()

```
template<typename Loss , typename T >
ceras::gradient\_descent< Loss, T >::gradient\_descent (
    Loss & loss,
    std::size_t batch_size,
    T learning_rate = 1.0e-3,
    T momentum = 0.0 ) [inline], [noexcept]
```

6.6.3 Member Function Documentation

6.6.3.1 forward()

```
template<typename Loss , typename T >
void ceras::gradient\_descent< Loss, T >::forward ( ) [inline]
```

6.6.4 Member Data Documentation

6.6.4.1 learning_rate_

```
template<typename Loss , typename T >
T ceras::gradient\_descent< Loss, T >::learning\_rate\_
```

6.6.4.2 loss_

```
template<typename Loss , typename T >
Loss& ceras::gradient_descent< Loss, T >::loss_
```

6.6.4.3 momentum_

```
template<typename Loss , typename T >
T ceras::gradient_descent< Loss, T >::momentum_
```

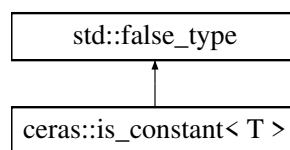
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/optimizer.hpp

6.7 ceras::is_constant< T > Struct Template Reference

```
#include <constant.hpp>
```

Inheritance diagram for ceras::is_constant< T >:



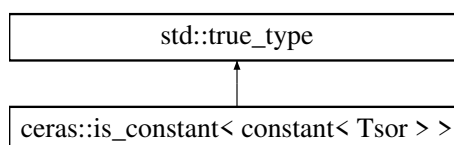
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/constant.hpp

6.8 ceras::is_constant< constant< Tsor > > Struct Template Reference

```
#include <constant.hpp>
```

Inheritance diagram for ceras::is_constant< constant< Tsor > >:



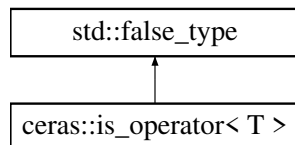
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/constant.hpp

6.9 ceras::is_operator< T > Struct Template Reference

```
#include <operation.hpp>
```

Inheritance diagram for ceras::is_operator< T >:



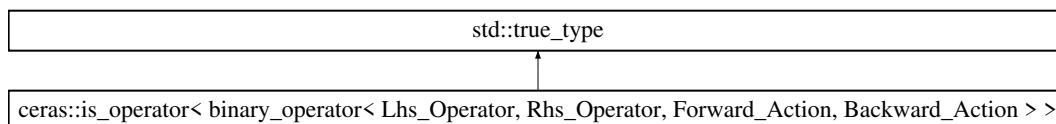
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/[operation.hpp](#)

6.10 ceras::is_operator< binary_operator< Lhs_Operator, Rh_Operator, Forward_Action, Backward_Action > > Struct Template Reference

```
#include <operation.hpp>
```

Inheritance diagram for ceras::is_operator< binary_operator< Lhs_Operator, Rh_Operator, Forward_Action, Backward_Action > >:



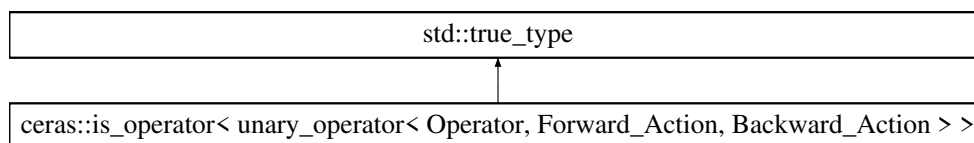
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/[operation.hpp](#)

6.11 ceras::is_operator< unary_operator< Operator, Forward_Action, Backward_Action > > Struct Template Reference

```
#include <operation.hpp>
```

Inheritance diagram for ceras::is_operator< unary_operator< Operator, Forward_Action, Backward_Action > >:



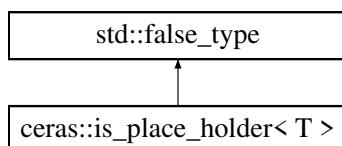
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/[operation.hpp](#)

6.12 ceras::is_place_holder< T > Struct Template Reference

```
#include <place_holder.hpp>
```

Inheritance diagram for ceras::is_place_holder< T >:



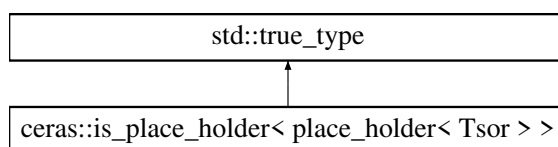
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/[place_holder.hpp](#)

6.13 ceras::is_place_holder< place_holder< Tsor > > Struct Template Reference

```
#include <place_holder.hpp>
```

Inheritance diagram for ceras::is_place_holder< place_holder< Tsor > >:



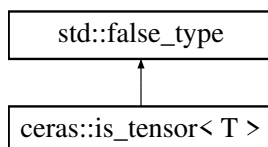
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/[place_holder.hpp](#)

6.14 ceras::is_tensor< T > Struct Template Reference

```
#include <tensor.hpp>
```

Inheritance diagram for ceras::is_tensor< T >:



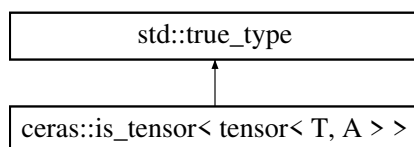
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/[tensor.hpp](#)

6.15 `ceras::is_tensor< tensor< T, A > >` Struct Template Reference

```
#include <tensor.hpp>
```

Inheritance diagram for `ceras::is_tensor< tensor< T, A > >`:



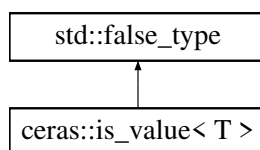
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/tensor.hpp

6.16 `ceras::is_value< T >` Struct Template Reference

```
#include <value.hpp>
```

Inheritance diagram for `ceras::is_value< T >`:



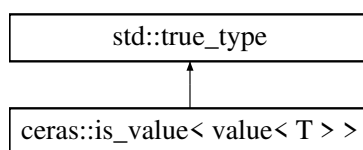
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/value.hpp

6.17 `ceras::is_value< value< T > >` Struct Template Reference

```
#include <value.hpp>
```

Inheritance diagram for `ceras::is_value< value< T > >`:



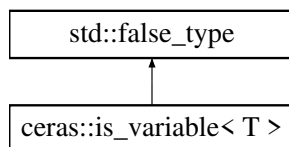
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/value.hpp

6.18 ceras::is_variable< T > Struct Template Reference

```
#include <variable.hpp>
```

Inheritance diagram for ceras::is_variable< T >:



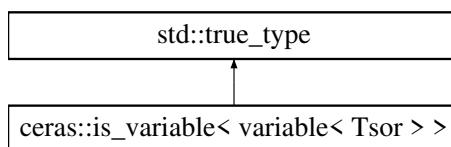
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/variable.hpp

6.19 ceras::is_variable< variable< Tsor > > Struct Template Reference

```
#include <variable.hpp>
```

Inheritance diagram for ceras::is_variable< variable< Tsor > >:



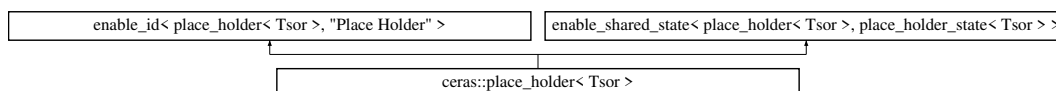
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/variable.hpp

6.20 ceras::place_holder< Tsor > Struct Template Reference

```
#include <place_holder.hpp>
```

Inheritance diagram for ceras::place_holder< Tsor >:



Public Types

- typedef Tsor [tensor_type](#)

Public Member Functions

- [place_holder](#) ([place_holder](#) const &other)=default
- [place_holder](#) ([place_holder](#) &&other)=default
- [place_holder](#) & [operator=](#) ([place_holder](#) const &other)=default
- [place_holder](#) & [operator=](#) ([place_holder](#) &&other)=default
- [place_holder](#) ()
- [place_holder](#) (std::vector< unsigned long > const &shape_hint)
- void [bind](#) (Tsor data)
- Tsor const [forward](#) () const
- void [reset](#) ()
- void [backward](#) (auto) const noexcept

6.20.1 Member Typedef Documentation

6.20.1.1 tensor_type

```
template<Tensor Tsor>
typedef Tsor ceras::place\_holder< Tsor >::tensor\_type
```

6.20.2 Constructor & Destructor Documentation

6.20.2.1 place_holder() [1/4]

```
template<Tensor Tsor>
ceras::place\_holder< Tsor >::place\_holder (
    place\_holder< Tsor > const & other ) [default]
```

6.20.2.2 place_holder() [2/4]

```
template<Tensor Tsor>
ceras::place\_holder< Tsor >::place\_holder (
    place\_holder< Tsor > && other ) [default]
```

6.20.2.3 place_holder() [3/4]

```
template<Tensor Tsor>
ceras::place\_holder< Tsor >::place\_holder ( ) [inline]
```

6.20.2.4 place_holder() [4/4]

```
template<Tensor Tsor>
ceras::place_holder< Tsor >::place_holder (
    std::vector< unsigned long > const & shape_hint ) [inline]
```

6.20.3 Member Function Documentation

6.20.3.1 backward()

```
template<Tensor Tsor>
void ceras::place_holder< Tsor >::backward (
    auto ) const [inline], [noexcept]
```

6.20.3.2 bind()

```
template<Tensor Tsor>
void ceras::place_holder< Tsor >::bind (
    Tsor data ) [inline]
```

6.20.3.3 forward()

```
template<Tensor Tsor>
Tsor const ceras::place_holder< Tsor >::forward ( ) const [inline]
```

6.20.3.4 operator=() [1/2]

```
template<Tensor Tsor>
place_holder& ceras::place_holder< Tsor >::operator= (
    place_holder< Tsor > && other ) [default]
```

6.20.3.5 operator=() [2/2]

```
template<Tensor Tsor>
place_holder& ceras::place_holder< Tsor >::operator= (
    place_holder< Tsor > const & other ) [default]
```


6.20.3.6 reset()

```
template<Tensor Tzor>
void ceras::place_holder< Tzor >::reset ( ) [inline]
```

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

- /data/structured_folders/workspace/github.repo/ceras/include/place_holder.hpp

6.21 ceras::place_holder_state< Tzor > Struct Template Reference

```
#include <place_holder.hpp>
```

Public Attributes

- Tzor [data_](#)
- std::vector< unsigned long > [shape_hint_](#)

6.21.1 Member Data Documentation

6.21.1.1 data_

```
template<Tensor Tzor>
Tzor ceras::place_holder_state< Tzor >::data_
```

6.21.1.2 shape_hint_

```
template<Tensor Tzor>
std::vector< unsigned long> ceras::place_holder_state< Tzor >::shape_hint_
```

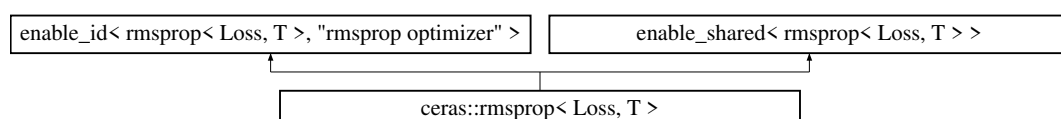
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/place_holder.hpp

6.22 ceras::rmsprop< Loss, T > Struct Template Reference

```
#include <optimizer.hpp>
```

Inheritance diagram for ceras::rmsprop< Loss, T >:



Public Types

- typedef [tensor](#)< T > [tensor_type](#)

Public Member Functions

- [rmsprop](#) (Loss &loss, std::size_t batch_size, T learning_rate=1.0e-1, T rho=0.9, T decay=0.0) noexcept
- void [forward](#) ()

Public Attributes

- Loss & [loss_](#)
- T [learning_rate_](#)
- T [rho_](#)
- T [decay_](#)
- unsigned long [iterations_](#)

6.22.1 Member Typedef Documentation

6.22.1.1 [tensor_type](#)

```
template<typename Loss , typename T >
typedef tensor< T > ceras::rmsprop< Loss, T >::tensor\_type
```

6.22.2 Constructor & Destructor Documentation

6.22.2.1 [rmsprop\(\)](#)

```
template<typename Loss , typename T >
ceras::rmsprop< Loss, T >::rmsprop (
    Loss & loss,
    std::size_t batch\_size,
    T learning\_rate = 1.0e-1,
    T rho = 0.9,
    T decay = 0.0 ) [inline], [noexcept]
```

6.22.3 Member Function Documentation

6.22.3.1 forward()

```
template<typename Loss , typename T >
void ceras::rmsprop< Loss, T >::forward ( ) [inline]
```

6.22.4 Member Data Documentation

6.22.4.1 decay_

```
template<typename Loss , typename T >
T ceras::rmsprop< Loss, T >::decay_
```

6.22.4.2 iterations_

```
template<typename Loss , typename T >
unsigned long ceras::rmsprop< Loss, T >::iterations_
```

6.22.4.3 learning_rate_

```
template<typename Loss , typename T >
T ceras::rmsprop< Loss, T >::learning_rate_
```

6.22.4.4 loss_

```
template<typename Loss , typename T >
Loss& ceras::rmsprop< Loss, T >::loss_
```

6.22.4.5 rho_

```
template<typename Loss , typename T >
T ceras::rmsprop< Loss, T >::rho_
```

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

- [/data/structured_folders/workspace/github.repo/ceras/include/optimizer.hpp](#)

6.23 ceras::session< Ttor > Struct Template Reference

```
#include <session.hpp>
```

Public Types

- typedef [place_holder](#)< Ttor > [place_holder_type](#)
- typedef [variable](#)< Ttor > [variable_type](#)
- typedef [variable_state](#)< Ttor > [variable_state_type](#)

Public Member Functions

- [session](#) ()
- [session](#) ([session](#) const &)=delete
- [session](#) ([session](#) &&)=delete
- [session](#) & [operator=](#) ([session](#) const &)=delete
- [session](#) & [operator=](#) ([session](#) &&)=delete
- void [rebind](#) ([place_holder_type](#) &p_holder, Ttor const &[value](#))
- void [bind](#) ([place_holder_type](#) &p_holder, Ttor const &[value](#))
- void [remember](#) ([variable_type](#) const &[v](#))
- template<typename Operation >
auto [run](#) (Operation &op) const
- template<typename Operation >
void [tap](#) (Operation &op) const
- void [deserialize](#) (std::string const &file_path)
- void [serialize](#) (std::string const &file_path) const
- void [save](#) (std::string const &file_path) const
- void [restore](#) (std::string const &file_path)
- [~session](#) ()

Public Attributes

- std::vector< [place_holder_type](#) > [place_holders_](#)
- std::unordered_map< int, [variable_type](#) > [variables_](#)

6.23.1 Member Typedef Documentation

6.23.1.1 place_holder_type

```
template<Tensor Ttor>
typedef place\_holder<Ttor> ceras::session< Ttor >::place\_holder\_type
```

6.23.1.2 variable_state_type

```
template<Tensor Tzor>
typedef variable_state<Tzor> ceras::session< Tzor >::variable_state_type
```

6.23.1.3 variable_type

```
template<Tensor Tzor>
typedef variable<Tzor> ceras::session< Tzor >::variable_type
```

6.23.2 Constructor & Destructor Documentation

6.23.2.1 session() [1/3]

```
template<Tensor Tzor>
ceras::session< Tzor >::session ( ) [inline]
```

6.23.2.2 session() [2/3]

```
template<Tensor Tzor>
ceras::session< Tzor >::session (
    session< Tzor > const & ) [delete]
```

6.23.2.3 session() [3/3]

```
template<Tensor Tzor>
ceras::session< Tzor >::session (
    session< Tzor > && ) [delete]
```

6.23.2.4 ~session()

```
template<Tensor Tzor>
ceras::session< Tzor >::~~session ( ) [inline]
```

6.23.3 Member Function Documentation

6.23.3.1 bind()

```
template<Tensor Tsor>
void ceras::session< Tsor >::bind (
    place_holder_type & p_holder,
    Tsor const & value ) [inline]
```

6.23.3.2 deserialize()

```
template<Tensor Tsor>
void ceras::session< Tsor >::deserialize (
    std::string const & file_path ) [inline]
```

6.23.3.3 operator=() [1/2]

```
template<Tensor Tsor>
session& ceras::session< Tsor >::operator= (
    session< Tsor > && ) [delete]
```

6.23.3.4 operator=() [2/2]

```
template<Tensor Tsor>
session& ceras::session< Tsor >::operator= (
    session< Tsor > const & ) [delete]
```

6.23.3.5 rebind()

```
template<Tensor Tsor>
void ceras::session< Tsor >::rebind (
    place_holder_type & p_holder,
    Tsor const & value ) [inline]
```

6.23.3.6 remember()

```
template<Tensor Tsor>
void ceras::session< Tsor >::remember (
    variable_type const & v ) [inline]
```

6.23.3.7 restore()

```
template<Tensor Tsor>
void ceras::session< Tsor >::restore (
    std::string const & file_path ) [inline]
```

6.23.3.8 run()

```
template<Tensor Tsor>
template<typename Operation >
auto ceras::session< Tsor >::run (
    Operation & op ) const [inline]
```

6.23.3.9 save()

```
template<Tensor Tsor>
void ceras::session< Tsor >::save (
    std::string const & file_path ) const [inline]
```

6.23.3.10 serialize()

```
template<Tensor Tsor>
void ceras::session< Tsor >::serialize (
    std::string const & file_path ) const [inline]
```

6.23.3.11 tap()

```
template<Tensor Tsor>
template<typename Operation >
void ceras::session< Tsor >::tap (
    Operation & op ) const [inline]
```

6.23.4 Member Data Documentation

6.23.4.1 place_holders_

```
template<Tensor Tsor>
std::vector<place_holder_type> ceras::session< Tsor >::place_holders_
```

6.23.4.2 variables_

```
template<Tensor Tsor>
std::unordered_map<int, variable_type> ceras::session< Tsor >::variables_
```

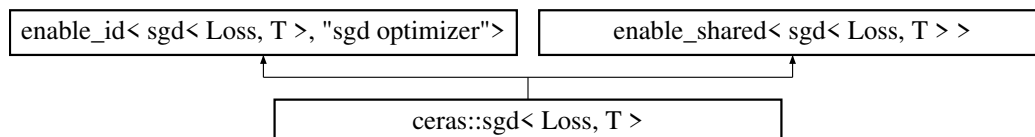
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/session.hpp

6.24 ceras::sgd< Loss, T > Struct Template Reference

```
#include <optimizer.hpp>
```

Inheritance diagram for ceras::sgd< Loss, T >:



Public Types

- typedef `tensor< T >` `tensor_type`

Public Member Functions

- `sgd` (Loss &loss, std::size_t batch_size, T learning_rate=1.0e-1, T momentum=0.0, T decay=0.0, bool nestorov=false) noexcept
- void `forward` ()

Public Attributes

- Loss & [loss_](#)
- T [learning_rate_](#)
- T [momentum_](#)
- T [decay_](#)
- bool [nesterov_](#)
- unsigned long [iterations_](#)

6.24.1 Member Typedef Documentation

6.24.1.1 `tensor_type`

```
template<typename Loss , typename T >  
typedef tensor< T > ceras::sgd< Loss, T >::tensor\_type
```

6.24.2 Constructor & Destructor Documentation

6.24.2.1 `sgd()`

```
template<typename Loss , typename T >  
ceras::sgd< Loss, T >::sgd (  
    Loss & loss,  
    std::size_t batch_size,  
    T learning_rate = 1.0e-1,  
    T momentum = 0.0,  
    T decay = 0.0,  
    bool nesterov = false ) [inline], [noexcept]
```

6.24.3 Member Function Documentation

6.24.3.1 `forward()`

```
template<typename Loss , typename T >  
void ceras::sgd< Loss, T >::forward ( ) [inline]
```

6.24.4 Member Data Documentation

6.24.4.1 decay_

```
template<typename Loss , typename T >
T ceras::sgd< Loss, T >::decay_
```

6.24.4.2 iterations_

```
template<typename Loss , typename T >
unsigned long ceras::sgd< Loss, T >::iterations_
```

6.24.4.3 learning_rate_

```
template<typename Loss , typename T >
T ceras::sgd< Loss, T >::learning_rate_
```

6.24.4.4 loss_

```
template<typename Loss , typename T >
Loss& ceras::sgd< Loss, T >::loss_
```

6.24.4.5 momentum_

```
template<typename Loss , typename T >
T ceras::sgd< Loss, T >::momentum_
```

6.24.4.6 nesterov_

```
template<typename Loss , typename T >
bool ceras::sgd< Loss, T >::nesterov_
```

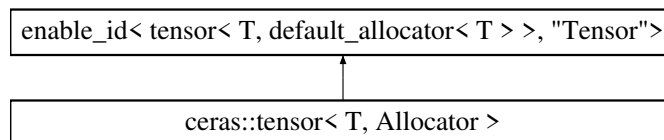
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/optimizer.hpp

6.25 ceras::tensor< T, Allocator > Struct Template Reference

```
#include <tensor.hpp>
```

Inheritance diagram for ceras::tensor< T, Allocator >:



Public Types

- typedef T [value_type](#)
- typedef Allocator [allocator](#)
- typedef std::vector< T, Allocator > [vector_type](#)
- typedef std::shared_ptr< [vector_type](#) > [shared_vector](#)
- typedef [tensor](#) [self_type](#)

Public Member Functions

- [tensor slice](#) (unsigned long m, unsigned long n) const noexcept
- constexpr auto [begin](#) () noexcept
- constexpr auto [begin](#) () const noexcept
- constexpr auto [cbegin](#) () const noexcept
- constexpr auto [end](#) () noexcept
- constexpr auto [end](#) () const noexcept
- constexpr auto [cend](#) () const noexcept
- constexpr [self_type](#) & [reset](#) (T val=T{0})
- constexpr unsigned long [ndim](#) () const noexcept
- constexpr [self_type](#) & [deep_copy](#) ([self_type](#) const &other)
- constexpr [self_type](#) const [deep_copy](#) () const
- constexpr [self_type](#) const [copy](#) () const
- constexpr [value_type](#) & [operator\[\]](#) (unsigned long idx)
- constexpr [value_type](#) const & [operator\[\]](#) (unsigned long idx) const
- [tensor](#) ()
- constexpr [tensor](#) (std::vector< unsigned long > const &[shape](#), std::initializer_list< T > init, const Allocator &alloc=Allocator())
- constexpr [tensor](#) (std::vector< unsigned long > const &[shape](#))
- constexpr [tensor](#) (std::vector< unsigned long > const &[shape](#), T init)
- constexpr [tensor](#) ([tensor](#) const &other, unsigned long memory_offset)
- constexpr [tensor](#) ([self_type](#) const &other) noexcept
- constexpr [tensor](#) ([self_type](#) &&other) noexcept
- constexpr [self_type](#) & [operator=](#) ([self_type](#) const &other) noexcept
- constexpr [self_type](#) & [operator=](#) ([self_type](#) &&other) noexcept
- constexpr std::vector< unsigned long > const & [shape](#) () const noexcept
- constexpr unsigned long [size](#) () const noexcept
- constexpr [self_type](#) & [resize](#) (std::vector< unsigned long > const &new_shape)
- constexpr [self_type](#) & [reshape](#) (std::vector< unsigned long > const &new_shape)
- constexpr [self_type](#) & [shrink_to](#) (std::vector< unsigned long > const &new_shape)
- constexpr [self_type](#) & [creep_to](#) (unsigned long new_memory_offset)

- constexpr bool `empty` () const noexcept
- constexpr `value_type` * `data` () noexcept
- constexpr const `value_type` * `data` () const noexcept
- template<typename Function >
constexpr `self_type` & `map` (Function const &f)
- constexpr `self_type` & `operator+=` (`self_type` const &other)
- constexpr `self_type` & `operator+=` (`value_type` x)
- constexpr `self_type` & `operator-=` (`self_type` const &other)
- constexpr `self_type` & `operator-=` (`value_type` x)
- constexpr `self_type` & `operator*=` (`self_type` const &other)
- constexpr `self_type` & `operator*=` (`value_type` x)
- constexpr `self_type` & `operator/=` (`self_type` const &other)
- constexpr `self_type` & `operator/=` (`value_type` x)
- constexpr `self_type` const `operator-` () const

Public Attributes

- std::vector< unsigned long > `shape_`
- unsigned long `memory_offset_`
- `shared_vector` `vector_`

6.25.1 Member Typedef Documentation

6.25.1.1 allocator

```
template<typename T , typename Allocator = default_allocator<T>>
typedef Allocator ceras::tensor< T, Allocator >::allocator
```

6.25.1.2 self_type

```
template<typename T , typename Allocator = default_allocator<T>>
typedef tensor ceras::tensor< T, Allocator >::self_type
```

6.25.1.3 shared_vector

```
template<typename T , typename Allocator = default_allocator<T>>
typedef std::shared_ptr<vector_type> ceras::tensor< T, Allocator >::shared_vector
```

6.25.1.4 value_type

```
template<typename T , typename Allocator = default_allocator<T>>
typedef T ceras::tensor< T, Allocator >::value_type
```

6.25.1.5 vector_type

```
template<typename T , typename Allocator = default_allocator<T>>
typedef std::vector<T, Allocator> ceras::tensor< T, Allocator >::vector_type
```

6.25.2 Constructor & Destructor Documentation

6.25.2.1 tensor() [1/7]

```
template<typename T , typename Allocator = default_allocator<T>>
ceras::tensor< T, Allocator >::tensor ( ) [inline]
```

6.25.2.2 tensor() [2/7]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr ceras::tensor< T, Allocator >::tensor (
    std::vector< unsigned long > const & shape,
    std::initializer_list< T > init,
    const Allocator & alloc = Allocator() ) [inline], [constexpr]
```

6.25.2.3 tensor() [3/7]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr ceras::tensor< T, Allocator >::tensor (
    std::vector< unsigned long > const & shape ) [inline], [constexpr]
```

6.25.2.4 tensor() [4/7]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr ceras::tensor< T, Allocator >::tensor (
    std::vector< unsigned long > const & shape,
    T init ) [inline], [constexpr]
```

6.25.2.5 tensor() [5/7]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr ceras::tensor< T, Allocator >::tensor (
    tensor< T, Allocator > const & other,
    unsigned long memory_offset ) [inline], [constexpr]
```

6.25.2.6 tensor() [6/7]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr ceras::tensor< T, Allocator >::tensor (
    self_type const & other ) [inline], [constexpr], [noexcept]
```

6.25.2.7 tensor() [7/7]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr ceras::tensor< T, Allocator >::tensor (
    self_type && other ) [inline], [constexpr], [noexcept]
```

6.25.3 Member Function Documentation**6.25.3.1 begin() [1/2]**

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr auto ceras::tensor< T, Allocator >::begin ( ) const [inline], [constexpr], [noexcept]
```

6.25.3.2 begin() [2/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr auto ceras::tensor< T, Allocator >::begin ( ) [inline], [constexpr], [noexcept]
```

6.25.3.3 cbegin()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr auto ceras::tensor< T, Allocator >::cbegin ( ) const [inline], [constexpr], [noexcept]
```

6.25.3.4 cend()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr auto ceras::tensor< T, Allocator >::cend ( ) const [inline], [constexpr], [noexcept]
```

6.25.3.5 copy()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type const ceras::tensor< T, Allocator >::copy ( ) const [inline], [constexpr]
```

6.25.3.6 creep_to()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::creep_to (
    unsigned long new_memory_offset ) [inline], [constexpr]
```

6.25.3.7 data() [1/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr const value_type* ceras::tensor< T, Allocator >::data ( ) const [inline], [constexpr],
[noexcept]
```

6.25.3.8 data() [2/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr value_type* ceras::tensor< T, Allocator >::data ( ) [inline], [constexpr], [noexcept]
```

6.25.3.9 deep_copy() [1/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type const ceras::tensor< T, Allocator >::deep_copy ( ) const [inline], [constexpr]
```

6.25.3.10 deep_copy() [2/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::deep_copy (
    self_type const & other ) [inline], [constexpr]
```

6.25.3.11 empty()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr bool ceras::tensor< T, Allocator >::empty ( ) const [inline], [constexpr], [noexcept]
```

6.25.3.12 end() [1/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr auto ceras::tensor< T, Allocator >::end ( ) const [inline], [constexpr], [noexcept]
```

6.25.3.13 end() [2/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr auto ceras::tensor< T, Allocator >::end ( ) [inline], [constexpr], [noexcept]
```

6.25.3.14 map()

```
template<typename T , typename Allocator = default_allocator<T>>
template<typename Function >
constexpr self_type& ceras::tensor< T, Allocator >::map (
    Function const & f ) [inline], [constexpr]
```

6.25.3.15 ndim()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr unsigned long ceras::tensor< T, Allocator >::ndim ( ) const [inline], [constexpr],
[noexcept]
```


6.25.3.16 operator*=() [1/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::operator*= (
    self_type const & other ) [inline], [constexpr]
```

6.25.3.17 operator*=() [2/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::operator*= (
    value_type x ) [inline], [constexpr]
```

6.25.3.18 operator+=() [1/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::operator+= (
    self_type const & other ) [inline], [constexpr]
```

6.25.3.19 operator+=() [2/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::operator+= (
    value_type x ) [inline], [constexpr]
```

6.25.3.20 operator-()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type const ceras::tensor< T, Allocator >::operator- ( ) const [inline], [constexpr]
```

6.25.3.21 operator-=() [1/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::operator-= (
    self_type const & other ) [inline], [constexpr]
```

6.25.3.22 operator-=() [2/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::operator-= (
    value_type x ) [inline], [constexpr]
```

6.25.3.23 operator/=() [1/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::operator/= (
    self_type const & other ) [inline], [constexpr]
```

6.25.3.24 operator/=() [2/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::operator/= (
    value_type x ) [inline], [constexpr]
```

6.25.3.25 operator=() [1/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::operator= (
    self_type && other ) [inline], [constexpr], [noexcept]
```

6.25.3.26 operator=() [2/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::operator= (
    self_type const & other ) [inline], [constexpr], [noexcept]
```

6.25.3.27 operator[]() [1/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr value_type& ceras::tensor< T, Allocator >::operator[] (
    unsigned long idx ) [inline], [constexpr]
```

6.25.3.28 operator[]() [2/2]

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr value_type const& ceras::tensor< T, Allocator >::operator[] (
    unsigned long idx ) const [inline], [constexpr]
```

6.25.3.29 reset()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::reset (
    T val = T{0} ) [inline], [constexpr]
```

6.25.3.30 reshape()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::reshape (
    std::vector< unsigned long > const & new_shape ) [inline], [constexpr]
```

6.25.3.31 resize()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::resize (
    std::vector< unsigned long > const & new_shape ) [inline], [constexpr]
```

6.25.3.32 shape()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr std::vector< unsigned long > const& ceras::tensor< T, Allocator >::shape ( ) const
[inline], [constexpr], [noexcept]
```

6.25.3.33 shrink_to()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr self_type& ceras::tensor< T, Allocator >::shrink_to (
    std::vector< unsigned long > const & new_shape ) [inline], [constexpr]
```

6.25.3.34 size()

```
template<typename T , typename Allocator = default_allocator<T>>
constexpr unsigned long ceras::tensor< T, Allocator >::size ( ) const [inline], [constexpr],
[noexcept]
```

6.25.3.35 slice()

```
template<typename T , typename Allocator = default_allocator<T>>
tensor ceras::tensor< T, Allocator >::slice (
    unsigned long m,
    unsigned long n ) const [inline], [noexcept]
```

6.25.4 Member Data Documentation

6.25.4.1 memory_offset_

```
template<typename T , typename Allocator = default_allocator<T>>
unsigned long ceras::tensor< T, Allocator >::memory_offset_
```

6.25.4.2 shape_

```
template<typename T , typename Allocator = default_allocator<T>>
std::vector<unsigned long> ceras::tensor< T, Allocator >::shape_
```

6.25.4.3 vector_

```
template<typename T , typename Allocator = default_allocator<T>>
shared_vector ceras::tensor< T, Allocator >::vector_
```

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

- /data/structured_folders/workspace/github.repo/ceras/include/[tensor.hpp](#)

6.26 ceras::tensor_deduction< L, R > Struct Template Reference

```
#include <value.hpp>
```

Public Types

- using `op_type` = `std::conditional< is_value_v< L >, R, L >::type`
- using `tensor_type` = `std::remove_cv_t< decltype(std::declval< op_type >()).forward())>`

6.26.1 Member Typedef Documentation

6.26.1.1 op_type

```
template<typename L , typename R >
using ceras::tensor_deduction< L, R >::op_type = std::conditional<is_value_v<L>, R, L>::type
```

6.26.1.2 tensor_type

```
template<typename L , typename R >
using ceras::tensor_deduction< L, R >::tensor_type = std::remove_cv_t<decltype(std::declval<op_type>()).forwa
```

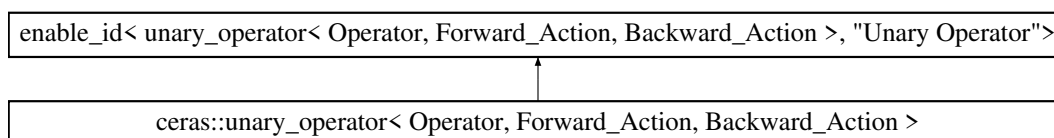
The documentation for this struct was generated from the following file:

- `/data/structured_folders/workspace/github.repo/ceras/include/value.hpp`

6.27 ceras::unary_operator< Operator, Forward_Action, Backward_Action > Struct Template Reference

```
#include <operation.hpp>
```

Inheritance diagram for `ceras::unary_operator< Operator, Forward_Action, Backward_Action >`:



Public Member Functions

- `unary_operator` (`Operator` const &op, `Forward_Action` const &forward_action, `Backward_Action` const &backward_action, `std::function< void()>` const &reset_action) noexcept
- auto `forward` ()
- void `backward` (`tensor_type` const &grad)
- void `reset_states` ()

Public Attributes

- [Operator](#) `op_`
- [Forward_Action](#) `forward_action_`
- [Backward_Action](#) `backward_action_`
- `std::function< void()>` `reset_action_`
- `decltype(std::declval< Forward_Action >().std::declval< decltype(op_)>().forward()))` typedef `tensor_type`
- `tensor_type` `input_data_`
- `tensor_type` `output_data_`

6.27.1 Constructor & Destructor Documentation

6.27.1.1 unary_operator()

```
template<typename Operator , typename Forward_Action , typename Backward_Action >
ceras::unary_operator< Operator, Forward_Action, Backward_Action >::unary_operator (
    Operator const & op,
    Forward_Action const & forward_action,
    Backward_Action const & backward_action,
    std::function< void()> const & reset_action ) [inline], [noexcept]
```

6.27.2 Member Function Documentation

6.27.2.1 backward()

```
template<typename Operator , typename Forward_Action , typename Backward_Action >
void ceras::unary_operator< Operator, Forward_Action, Backward_Action >::backward (
    tensor_type const & grad ) [inline]
```

6.27.2.2 forward()

```
template<typename Operator , typename Forward_Action , typename Backward_Action >
auto ceras::unary_operator< Operator, Forward_Action, Backward_Action >::forward ( ) [inline]
```

6.27.2.3 reset_states()

```
template<typename Operator , typename Forward_Action , typename Backward_Action >
void ceras::unary_operator< Operator, Forward_Action, Backward_Action >::reset_states ( )
[inline]
```

6.27.3 Member Data Documentation

6.27.3.1 backward_action_

```
template<typename Operator , typename Forward_Action , typename Backward_Action >  
Backward_Action ceras::unary_operator< Operator, Forward_Action, Backward_Action >::backward_  
_action_
```

6.27.3.2 forward_action_

```
template<typename Operator , typename Forward_Action , typename Backward_Action >  
Forward_Action ceras::unary_operator< Operator, Forward_Action, Backward_Action >::forward_  
action_
```

6.27.3.3 input_data_

```
template<typename Operator , typename Forward_Action , typename Backward_Action >  
tensor_type ceras::unary_operator< Operator, Forward_Action, Backward_Action >::input_data_
```

6.27.3.4 op_

```
template<typename Operator , typename Forward_Action , typename Backward_Action >  
Operator ceras::unary_operator< Operator, Forward_Action, Backward_Action >::op_
```

6.27.3.5 output_data_

```
template<typename Operator , typename Forward_Action , typename Backward_Action >  
tensor_type ceras::unary_operator< Operator, Forward_Action, Backward_Action >::output_data_
```

6.27.3.6 reset_action_

```
template<typename Operator , typename Forward_Action , typename Backward_Action >  
std::function<void()> ceras::unary_operator< Operator, Forward_Action, Backward_Action >::  
::reset_action_
```

6.27.3.7 tensor_type

```
template<typename Operator , typename Forward_Action , typename Backward_Action >
decltype( std::declval<Forward_Action>() ( std::declval<decltype(op_)>().forward() ) ) typedef
ceras::unary_operator< Operator, Forward_Action, Backward_Action >::tensor_type
```

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

- [/data/structured_folders/workspace/github.repo/ceras/include/operation.hpp](#)

6.28 ceras::value< T > Struct Template Reference

```
#include <value.hpp>
```

Public Types

- typedef T [value_type](#)

Public Member Functions

- [value](#) ()=delete
- [value](#) ([value_type](#) v) noexcept
- [value](#) ([value](#) const &) noexcept=default
- [value](#) ([value](#) &&) noexcept=default
- [value](#) & [operator=](#) ([value](#) const &) noexcept=default
- [value](#) & [operator=](#) ([value](#) &&) noexcept=default
- void [backward](#) (auto) noexcept
- template<Tensor Tsor>
Tsor const [forward](#) (Tsor const &refer) const

Public Attributes

- [value_type](#) [data_](#)

6.28.1 Member Typedef Documentation

6.28.1.1 value_type

```
template<typename T >
typedef T ceras::value< T >::value_type
```


6.28.2 Constructor & Destructor Documentation

6.28.2.1 value() [1/4]

```
template<typename T >
ceras::value< T >::value ( ) [delete]
```

6.28.2.2 value() [2/4]

```
template<typename T >
ceras::value< T >::value (
    value_type v ) [inline], [noexcept]
```

6.28.2.3 value() [3/4]

```
template<typename T >
ceras::value< T >::value (
    value< T > const & ) [default], [noexcept]
```

6.28.2.4 value() [4/4]

```
template<typename T >
ceras::value< T >::value (
    value< T > && ) [default], [noexcept]
```

6.28.3 Member Function Documentation

6.28.3.1 backward()

```
template<typename T >
void ceras::value< T >::backward (
    auto ) [inline], [noexcept]
```

6.28.3.2 forward()

```
template<typename T >
template<Tensor Tsor>
Tsor const ceras::value< T >::forward (
    Tsor const & refer ) const [inline]
```

6.28.3.3 operator=() [1/2]

```
template<typename T >
value& ceras::value< T >::operator= (
    value< T > && ) [default], [noexcept]
```

6.28.3.4 operator=() [2/2]

```
template<typename T >
value& ceras::value< T >::operator= (
    value< T > const & ) [default], [noexcept]
```

6.28.4 Member Data Documentation

6.28.4.1 data_

```
template<typename T >
value_type ceras::value< T >::data_
```

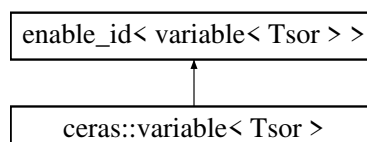
The documentation for this struct was generated from the following file:

- /data/structured_folders/workspace/github.repo/ceras/include/value.hpp

6.29 ceras::variable< Tsor > Struct Template Reference

```
#include <variable.hpp>
```

Inheritance diagram for ceras::variable< Tsor >:



Public Types

- typedef T_{sr} [tensor_type](#)

Public Member Functions

- [variable](#) (T_{sr} const &[data](#), bool trainable=true, bool stateful=false)
- [variable](#) ()=delete
- [variable](#) ([variable](#) const &other)=default
- [variable](#) ([variable](#) &&)=default
- [variable](#) & [operator=](#) ([variable](#) &&)=default
- [variable](#) & [operator=](#) ([variable](#) const &other)=default
- T_{sr} const [forward](#) () const
- void [backward](#) (auto const &grad)
- std::vector< std::size_t > [shape](#) () const noexcept
- T_{sr} & [data](#) ()
- T_{sr} [data](#) () const
- T_{sr} & [gradient](#) ()
- T_{sr} [gradient](#) () const
- void [reset](#) ()
- void [reset_states](#) ()

Public Attributes

- std::shared_ptr< [variable_state](#)< T_{sr} > > [state_](#)
- bool [trainable_](#)
- bool [stateful_](#)

6.29.1 Member Typedef Documentation

6.29.1.1 [tensor_type](#)

```
template<Tensor Tsr>
typedef Tsr ceras::variable< Tsr >::tensor\_type
```

6.29.2 Constructor & Destructor Documentation

6.29.2.1 [variable\(\)](#) [1/4]

```
template<Tensor Tsr>
ceras::variable< Tsr >::variable (
    Tsr const & data,
    bool trainable = true,
    bool stateful = false ) [inline]
```

6.29.2.2 `variable()` [2/4]

```
template<Tensor Tsor>
ceras::variable< Tsor >::variable ( ) [delete]
```

6.29.2.3 `variable()` [3/4]

```
template<Tensor Tsor>
ceras::variable< Tsor >::variable (
    variable< Tsor > const & other ) [default]
```

6.29.2.4 `variable()` [4/4]

```
template<Tensor Tsor>
ceras::variable< Tsor >::variable (
    variable< Tsor > && ) [default]
```

6.29.3 Member Function Documentation

6.29.3.1 `backward()`

```
template<Tensor Tsor>
void ceras::variable< Tsor >::backward (
    auto const & grad ) [inline]
```

6.29.3.2 `data()` [1/2]

```
template<Tensor Tsor>
Tsor& ceras::variable< Tsor >::data ( ) [inline]
```

6.29.3.3 `data()` [2/2]

```
template<Tensor Tsor>
Tsor ceras::variable< Tsor >::data ( ) const [inline]
```

6.29.3.4 forward()

```
template<Tensor Tzor>
Tzor const ceras::variable< Tzor >::forward ( ) const [inline]
```

6.29.3.5 gradient() [1/2]

```
template<Tensor Tzor>
Tzor& ceras::variable< Tzor >::gradient ( ) [inline]
```

6.29.3.6 gradient() [2/2]

```
template<Tensor Tzor>
Tzor ceras::variable< Tzor >::gradient ( ) const [inline]
```

6.29.3.7 operator=() [1/2]

```
template<Tensor Tzor>
variable& ceras::variable< Tzor >::operator= (
    variable< Tzor > && ) [default]
```

6.29.3.8 operator=() [2/2]

```
template<Tensor Tzor>
variable& ceras::variable< Tzor >::operator= (
    variable< Tzor > const & other ) [default]
```

6.29.3.9 reset()

```
template<Tensor Tzor>
void ceras::variable< Tzor >::reset ( ) [inline]
```

6.29.3.10 reset_states()

```
template<Tensor Tzor>
void ceras::variable< Tzor >::reset_states ( ) [inline]
```

6.29.3.11 shape()

```
template<Tensor Tsor>
std::vector<std::size_t> ceras::variable< Tsor >::shape ( ) const [inline], [noexcept]
```

6.29.4 Member Data Documentation

6.29.4.1 state_

```
template<Tensor Tsor>
std::shared_ptr<variable_state<Tsor> > ceras::variable< Tsor >::state_
```

6.29.4.2 stateful_

```
template<Tensor Tsor>
bool ceras::variable< Tsor >::stateful_
```

6.29.4.3 trainable_

```
template<Tensor Tsor>
bool ceras::variable< Tsor >::trainable_
```

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

- /data/structured_folders/workspace/github.repo/ceras/include/variable.hpp

6.30 ceras::variable_state< Tsor > Struct Template Reference

```
#include <variable.hpp>
```

Public Attributes

- Tsor [data_](#)
- Tsor [gradient_](#)

6.30.1 Member Data Documentation

6.30.1.1 data_

```
template<Tensor Tsor>
Tsor ceras::variable_state< Tsor >::data_
```

6.30.1.2 gradient_

```
template<Tensor Tsor>
Tsor ceras::variable_state< Tsor >::gradient_
```

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

- /data/structured_folders/workspace/github.repo/ceras/include/[variable.hpp](#)

6.31 ceras::view_2d< T > Struct Template Reference

```
#include <tensor.hpp>
```

Public Member Functions

- template<typename A >
constexpr [view_2d](#) ([tensor](#)< T, A > &tsor, unsigned long row, unsigned long col, bool transposed=false) noexcept
- constexpr [view_2d](#) (T *[data](#), unsigned long row, unsigned long col, bool transposed=false) noexcept
- constexpr [view_2d](#) (const T *[data](#), unsigned long row, unsigned long col, bool transposed=false) noexcept
- constexpr T * [operator\[\]](#) (unsigned long index)
- constexpr const T * [operator\[\]](#) (unsigned long index) const
- constexpr auto [shape](#) () const noexcept
- constexpr unsigned long [size](#) () const noexcept
- constexpr T * [data](#) () noexcept
- constexpr const T * [data](#) () const noexcept

Public Attributes

- T * [data_](#)
- unsigned long [row_](#)
- unsigned long [col_](#)
- bool [transposed_](#)

6.31.1 Constructor & Destructor Documentation

6.31.1.1 view_2d() [1/3]

```
template<typename T >
template<typename A >
constexpr ceras::view_2d< T >::view_2d (
    tensor< T, A > & tsor,
    unsigned long row,
    unsigned long col,
    bool transposed = false ) [inline], [constexpr], [noexcept]
```

6.31.1.2 view_2d() [2/3]

```
template<typename T >
constexpr ceras::view_2d< T >::view_2d (
    T * data,
    unsigned long row,
    unsigned long col,
    bool transposed = false ) [inline], [constexpr], [noexcept]
```

6.31.1.3 view_2d() [3/3]

```
template<typename T >
constexpr ceras::view_2d< T >::view_2d (
    const T * data,
    unsigned long row,
    unsigned long col,
    bool transposed = false ) [inline], [constexpr], [noexcept]
```

6.31.2 Member Function Documentation**6.31.2.1 data() [1/2]**

```
template<typename T >
constexpr const T* ceras::view_2d< T >::data ( ) const [inline], [constexpr], [noexcept]
```

6.31.2.2 data() [2/2]

```
template<typename T >
constexpr T* ceras::view_2d< T >::data ( ) [inline], [constexpr], [noexcept]
```


6.31.2.3 operator[]() [1/2]

```
template<typename T >
constexpr T* ceras::view_2d< T >::operator[] (
    unsigned long index ) [inline], [constexpr]
```

6.31.2.4 operator[]() [2/2]

```
template<typename T >
constexpr const T* ceras::view_2d< T >::operator[] (
    unsigned long index ) const [inline], [constexpr]
```

6.31.2.5 shape()

```
template<typename T >
constexpr auto ceras::view_2d< T >::shape ( ) const [inline], [constexpr], [noexcept]
```

6.31.2.6 size()

```
template<typename T >
constexpr unsigned long ceras::view_2d< T >::size ( ) const [inline], [constexpr], [noexcept]
```

6.31.3 Member Data Documentation

6.31.3.1 col_

```
template<typename T >
unsigned long ceras::view_2d< T >::col_
```

6.31.3.2 data_

```
template<typename T >
T* ceras::view_2d< T >::data_
```

6.31.3.3 row_

```
template<typename T >
unsigned long ceras::view_2d< T >::row_
```

6.31.3.4 transposed_

```
template<typename T >
bool ceras::view_2d< T >::transposed_
```

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

- /data/structured_folders/workspace/github.repo/ceras/include/tensor.hpp

6.32 ceras::view_3d< T > Struct Template Reference

```
#include <tensor.hpp>
```

Public Member Functions

- constexpr [view_3d](#) (T *data, unsigned long row, unsigned long col, unsigned long channel) noexcept
- constexpr auto [operator\[\]](#) (unsigned long index) noexcept
- constexpr auto [operator\[\]](#) (unsigned long index) const noexcept

Public Attributes

- T * [data_](#)
- unsigned long [row_](#)
- unsigned long [col_](#)
- unsigned long [channel_](#)

6.32.1 Constructor & Destructor Documentation

6.32.1.1 view_3d()

```
template<typename T >
constexpr ceras::view_3d< T >::view_3d (
    T * data,
    unsigned long row,
    unsigned long col,
    unsigned long channel ) [inline], [constexpr], [noexcept]
```

6.32.2 Member Function Documentation

6.32.2.1 operator[]() [1/2]

```
template<typename T >
constexpr auto ceras::view_3d< T >::operator[] (
    unsigned long index ) const [inline], [constexpr], [noexcept]
```

6.32.2.2 operator[]() [2/2]

```
template<typename T >
constexpr auto ceras::view_3d< T >::operator[] (
    unsigned long index ) [inline], [constexpr], [noexcept]
```

6.32.3 Member Data Documentation

6.32.3.1 channel_

```
template<typename T >
unsigned long ceras::view_3d< T >::channel_
```

6.32.3.2 col_

```
template<typename T >
unsigned long ceras::view_3d< T >::col_
```

6.32.3.3 data_

```
template<typename T >
T* ceras::view_3d< T >::data_
```

6.32.3.4 row_

```
template<typename T >
unsigned long ceras::view_3d< T >::row_
```

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

- /data/structured_folders/workspace/github.repo/ceras/include/tensor.hpp

6.33 ceras::view_4d< T > Struct Template Reference

```
#include <tensor.hpp>
```

Public Member Functions

- constexpr [view_4d](#) (T *data, unsigned long batch_size, unsigned long row, unsigned long col, unsigned long channel) noexcept
- constexpr auto [operator\[\]](#) (unsigned long index) noexcept
- constexpr auto [operator\[\]](#) (unsigned long index) const noexcept

Public Attributes

- T * [data_](#)
The pointer to the start position of the 1-D array.
- unsigned long [batch_size_](#)
The batch size of the 4-D tensor, also the first dimension of the tensor.
- unsigned long [row_](#)
The row of the 4-D tensor, also the second dimension of the tensor.
- unsigned long [col_](#)
The column of the 4-D tensor, also the third dimension of the tensor.
- unsigned long [channel_](#)
The channel of the 4-D tensor, also the last dimension of the tensor.

6.33.1 Detailed Description

```
template<typename T>
struct ceras::view_4d< T >
```

A class viewing a 1-D array as a 4-D tensor. This class is useful when treating an array as a typical 4-D tensor in a neural network, with a shape of [batch_size, row, column, channel].

6.33.2 Constructor & Destructor Documentation

6.33.2.1 `view_4d()`

```
template<typename T >
constexpr ceras::view_4d< T >::view_4d (
    T * data,
    unsigned long batch_size,
    unsigned long row,
    unsigned long col,
    unsigned long channel ) [inline], [constexpr], [noexcept]
```

Constructor of `view_4d`

Parameters

<i>data</i>	The raw pointer to the start position of the 1-D array.
<i>batch_size</i>	The first dimension of the 4-D tensor, also for the batch size in the CNN layers.
<i>row</i>	The second dimension of the 4-D tensor, also for the row in the CNN layers.
<i>col</i>	The third dimension of the 4-D tensor, also for the column in the CNN layers.
<i>channel</i>	The last dimension of the 4-D tensor, also for the channel in the CNN layers.

6.33.3 Member Function Documentation

6.33.3.1 operator[]() [1/2]

```
template<typename T >
constexpr auto ceras::view_4d< T >::operator[] (
    unsigned long index ) const [inline], [constexpr], [noexcept]
```

Giving a [view_3d](#) interface for operator [].

Parameters

<i>index</i>	The first dimension of the 4-D tensor.
--------------	----------------------------------------

Example usage:

```
std::vector<float> array;
array.resize( 16*8*8*3 );
// operations on 'array'
auto t = view_4d{ array.data(), 16, 8, 8, 3 };
float v0123 = t[0][1][2][3];
```

6.33.3.2 operator[]() [2/2]

```
template<typename T >
constexpr auto ceras::view_4d< T >::operator[] (
    unsigned long index ) [inline], [constexpr], [noexcept]
```

Giving a [view_3d](#) interface for operator [].

Parameters

<i>index</i>	The first dimension of the 4-D tensor.
--------------	----------------------------------------

Example usage:

```
std::vector<float> array;
array.resize( 16*8*8*3 );
auto t = view_4d{ array.data(), 16, 8, 8, 3 };
t[0][1][2][3] = 1.0;
```

6.33.4 Member Data Documentation

6.33.4.1 batch_size_

```
template<typename T >
unsigned long ceras::view_4d< T >::batch_size_
```

The batch size of the 4-D tensor, also the first dimension of the tensor.

6.33.4.2 channel_

```
template<typename T >
unsigned long ceras::view_4d< T >::channel_
```

The channel of the 4-D tensor, also the last dimension of the tensor.

6.33.4.3 col_

```
template<typename T >
unsigned long ceras::view_4d< T >::col_
```

The column of the 4-D tensor, also the third dimension of the tensor.

6.33.4.4 data_

```
template<typename T >
T* ceras::view_4d< T >::data_
```

The pointer to the start position of the 1-D array.

6.33.4.5 row_

```
template<typename T >
unsigned long ceras::view_4d< T >::row_
```

The row of the 4-D tensor, also the second dimension of the tensor.

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

- [/data/structured_folders/workspace/github.repo/ceras/include/tensor.hpp](#)

Chapter 7

File Documentation

7.1 `/data/structured_↵` `folders/workspace/github.repo/ceras/include/activation.hpp` File Reference

```
#include "../operation.hpp"
#include "../tensor.hpp"
#include "../utils/range.hpp"
#include "../utils/better_assert.hpp"
#include "../utils/for_each.hpp"
#include "../utils/context_cast.hpp"
```

Namespaces

- [ceras](#)

Functions

- `template<Expression Ex>`
`constexpr auto ceras::softmax` (`Ex const &ex`) `noexcept`
- `template<Expression Ex>`
`auto ceras::selu` (`Ex const &ex`) `noexcept`
- `template<Expression Ex>`
`auto ceras::softplus` (`Ex const &ex`) `noexcept`
- `template<Expression Ex>`
`auto ceras::softsign` (`Ex const &ex`) `noexcept`
- `template<Expression Ex>`
`auto ceras::sigmoid` (`Ex const &ex`) `noexcept`
- `template<Expression Ex>`
`auto ceras::tanh` (`Ex const &ex`) `noexcept`
- `template<Expression Ex>`
`auto ceras::relu` (`Ex const &ex`) `noexcept`
- `template<typename T >`
`requires std::floating_point< T >` `auto ceras::leaky_relu` (`T const factor`) `noexcept`

- `template<typename T >`
`requires std::floating_point< T >` `auto ceras::elu` (T const alpha) noexcept
- `template<Expression Ex>`
`auto ceras::exponential` (Ex const &ex) noexcept
- `template<Expression Ex>`
`auto ceras::hard_sigmoid` (Ex const &ex) noexcept
- `template<Expression Ex>`
`auto ceras::gelu` (Ex const &ex) noexcept

7.2 /data/structured_↵

folders/workspace/github.repo/ceras/include/ceras.hpp File Reference

```
#include " ./config.hpp"
#include " ./includes.hpp"
#include " ./activation.hpp"
#include " ./ceras.hpp"
#include " ./loss.hpp"
#include " ./operation.hpp"
#include " ./optimizer.hpp"
#include " ./place_holder.hpp"
#include " ./session.hpp"
#include " ./tensor.hpp"
#include " ./variable.hpp"
#include " ./constant.hpp"
#include " ./layer.hpp"
```

7.3 /data/structured_↵

folders/workspace/github.repo/ceras/include/config.hpp File Reference

Namespaces

- `ceras`

Variables

- `constexpr unsigned long ceras::version` = 20210418UL
- `constexpr unsigned long ceras::__version__` = version
- `constexpr unsigned long ceras::is_windows_platform` = 0
- `constexpr unsigned long ceras::debug_mode` = 1
- `constexpr unsigned long ceras::blas_mode` = 0
- `constexpr unsigned long ceras::cuda_mode` = 0
- `int ceras::visible_device` = 0
- `unsigned long ceras::cuda_gemm_threshold` = 0UL
- `constexpr double ceras::eps` = 1.0e-8
- `int ceras::learning_phase` = 1

7.4 /data/structured_folders/workspace/github.repo/ceras/include/constant.hpp File Reference

```
#include "../includes.hpp"
#include "../tensor.hpp"
#include "../utils/id.hpp"
#include "../utils/better_assert.hpp"
#include "../utils/enable_shared.hpp"
```

Classes

- struct [ceras::constant< Tsor >](#)
- struct [ceras::is_constant< T >](#)
- struct [ceras::is_constant< constant< Tsor > >](#)

Namespaces

- [ceras](#)

Variables

- template<class T >
constexpr bool [ceras::is_constant_v](#) = is_constant<T>::value
- template<typename T >
concept [ceras::Constant](#) = is_constant_v<T>

7.5 /data/structured_folders/workspace/github.repo/ceras/include/includes.hpp File Reference

```
#include "../config.hpp"
#include <algorithm>
#include <any>
#include <array>
#include <cassert>
#include <chrono>
#include <cmath>
#include <compare>
#include <concepts>
#include <cstdint>
#include <ctime>
#include <filesystem>
#include <fstream>
#include <functional>
#include <initializer_list>
#include <iomanip>
#include <iostream>
```

```
#include <iterator>
#include <limits>
#include <map>
#include <memory>
#include <numeric>
#include <optional>
#include <ostream>
#include <random>
#include <regex>
#include <set>
#include <sstream>
#include <string>
#include <tuple>
#include <thread>
#include <type_traits>
#include <unordered_map>
#include <unordered_set>
#include <utility>
#include <vector>
#include "../utils/3rd_party/stb_image.h"
#include "../utils/3rd_party/stb_image_write.h"
#include "../utils/3rd_party/stb_image_resize.h"
#include "../utils/3rd_party/glob.hpp"
```

Macros

- `#define` [STB_IMAGE_IMPLEMENTATION](#)
- `#define` [STB_IMAGE_WRITE_IMPLEMENTATION](#)
- `#define` [STB_IMAGE_RESIZE_IMPLEMENTATION](#)

7.5.1 Macro Definition Documentation

7.5.1.1 STB_IMAGE_IMPLEMENTATION

```
#define STB_IMAGE_IMPLEMENTATION
```

7.5.1.2 STB_IMAGE_RESIZE_IMPLEMENTATION

```
#define STB_IMAGE_RESIZE_IMPLEMENTATION
```

7.5.1.3 STB_IMAGE_WRITE_IMPLEMENTATION

```
#define STB_IMAGE_WRITE_IMPLEMENTATION
```

7.6 /data/structured_↵/folders/workspace/github.repo/ceras/include/keras.hpp File Reference

```
#include "../keras/layer.hpp"
#include "../keras/activation.hpp"
#include "../keras/application.hpp"
#include "../keras/callback.hpp"
#include "../keras/constraint.hpp"
#include "../keras/dataset.hpp"
#include "../keras/initializer.hpp"
#include "../keras/loss.hpp"
#include "../keras/metric.hpp"
#include "../keras/model.hpp"
#include "../keras/optimizer.hpp"
#include "../keras/regularizer.hpp"
#include "../keras/visualization.hpp"
```

7.7 /data/structured_↵/folders/workspace/github.repo/ceras/include/layer.hpp File Reference

```
#include "../operation.hpp"
#include "../utils/better_assert.hpp"
```

Namespaces

- [ceras](#)

Functions

- auto [ceras::Input](#) ()
- auto [ceras::Conv2D](#) (unsigned long output_channels, std::vector< unsigned long > const &kernel_size, std::vector< unsigned long > const &input_shape, std::string const &padding="valid", std::vector< unsigned long > const &strides={1, 1})
- auto [ceras::Dense](#) (unsigned long output_size, unsigned long input_size)
- auto [ceras::BatchNormalization](#) (std::vector< unsigned long > const &shape, float threshold=0.95f)

7.8 /data/structured_↵/folders/workspace/github.repo/ceras/include/loss.hpp File Reference

```
#include "../operation.hpp"
#include "../tensor.hpp"
#include "../utils/debug.hpp"
```

Namespaces

- [ceras](#)

Functions

- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto ceras::mean_squared_logarithmic_error (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto ceras::squared_loss (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto ceras::mean_squared_error (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto ceras::mse (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto ceras::abs_loss (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto ceras::mean_absolute_error (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto ceras::mae (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto ceras::cross_entropy (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto ceras::cross_entropy_loss (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`
- `template<Expression Lhs_Expression, Expression Rhs_Expression>`
`constexpr auto ceras::hinge_loss (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept`

7.9 `/data/structured_↵`

`folders/workspace/github.repo/ceras/include/operation.hpp` File Reference

```
#include "../includes.hpp"
#include "../place_holder.hpp"
#include "../variable.hpp"
#include "../constant.hpp"
#include "../value.hpp"
#include "../utils/range.hpp"
#include "../utils/debug.hpp"
#include "../config.hpp"
#include "../utils/context_cast.hpp"
#include "../utils/for_each.hpp"
#include "../utils/id.hpp"
#include "../utils/enable_shared.hpp"
```

Classes

- struct [ceras::unary_operator< Operator, Forward_Action, Backward_Action >](#)
- struct [ceras::binary_operator< Lhs_Operator, Rhc_Operator, Forward_Action, Backward_Action >](#)
- struct [ceras::is_operator< T >](#)
- struct [ceras::is_operator< binary_operator< Lhs_Operator, Rhc_Operator, Forward_Action, Backward_Action > >](#)
- struct [ceras::is_operator< unary_operator< Operator, Forward_Action, Backward_Action > >](#)

Namespaces

- [ceras](#)

Functions

- template<Expression Lhs_Expression, Expression Rhc_Expression>
constexpr auto [ceras::plus](#) (Lhs_Expression const &lhc_ex, Rhc_Expression const &rhc_ex) noexcept
- template<Expression Lhs_Expression, Expression Rhc_Expression>
constexpr auto [ceras::operator+](#) (Lhs_Expression const &lhc_ex, Rhc_Expression const &rhc_ex) noexcept
- template<Expression Lhs_Expression, Expression Rhc_Expression>
auto [ceras::operator*](#) (Lhs_Expression const &lhc_ex, Rhc_Expression const &rhc_ex) noexcept
- template<Expression Ex>
constexpr auto [ceras::log](#) (Ex const &ex) noexcept
- template<Expression Ex>
constexpr auto [ceras::negative](#) (Ex const &ex) noexcept
- template<Expression Lhs_Expression, Expression Rhc_Expression>
constexpr auto [ceras::elementwise_product](#) (Lhs_Expression const &lhc_ex, Rhc_Expression const &rhc_ex) noexcept
- template<Expression Lhs_Expression, Expression Rhc_Expression>
constexpr auto [ceras::hadamard_product](#) (Lhs_Expression const &lhc_ex, Rhc_Expression const &rhc_ex) noexcept
- template<Expression Ex>
constexpr auto [ceras::sum_reduce](#) (Ex const &ex) noexcept
- template<Expression Ex>
constexpr auto [ceras::reduce_sum](#) (Ex const &ex) noexcept
- template<Expression Ex>
constexpr auto [ceras::mean_reduce](#) (Ex const &ex) noexcept
- template<Expression Ex>
constexpr auto [ceras::reduce_mean](#) (Ex const &ex) noexcept
- template<Expression Lhs_Expression, Expression Rhc_Expression>
constexpr auto [ceras::minus](#) (Lhs_Expression const &lhc_ex, Rhc_Expression const &rhc_ex) noexcept
- template<Expression Lhs_Expression, Expression Rhc_Expression>
constexpr auto [ceras::operator-](#) (Lhs_Expression const &lhc_ex, Rhc_Expression const &rhc_ex) noexcept
- template<Expression Ex>
constexpr auto [ceras::square](#) (Ex const &ex) noexcept
- template<Expression Ex>
constexpr auto [ceras::abs](#) (Ex const &ex) noexcept
- template<Expression Ex>
constexpr auto [ceras::exp](#) (Ex const &ex) noexcept
- template<typename Float >
requires std::floating_point< Float > constexpr auto [ceras::clip](#) (Float lower, Float upper=std::numeric_limits< Float >::max()) noexcept
- auto [ceras::reshape](#) (std::vector< unsigned long > const &new_shape, bool include_batch_flag=true) noexcept

- template<Expression Ex>
constexpr auto [ceras::flatten](#) (Ex const &ex) noexcept
- template<Expression Ex>
constexpr auto [ceras::identity](#) (Ex const &ex) noexcept
- template<Expression Ex>
auto [ceras::transpose](#) (Ex const &ex) noexcept
- auto [ceras::img2col](#) (unsigned long const row_kernel, unsigned long col_kernel=-1, unsigned long const row_padding=0, unsigned long col_padding=0, unsigned long const row_stride=1, unsigned long const col_stride=1, unsigned long const row_dilation=1, unsigned long const col_dilation=1) noexcept
- auto [ceras::conv2d](#) (unsigned long row_input, unsigned long col_input, unsigned long const row_stride=1, unsigned long const col_stride=1, unsigned long const row_dilation=1, unsigned long const col_dilation=1, std::string const &padding="valid") noexcept
- template<typename T >
requires std::floating_point< T > auto [ceras::drop_out](#) (T const factor) noexcept
- auto [ceras::max_pooling_2d](#) (unsigned long stride) noexcept
- auto [ceras::average_pooling_2d](#) (unsigned long stride) noexcept
- auto [ceras::up_sampling_2d](#) (unsigned long stride) noexcept
- template<typename T = double>
requires std::floating_point< T > auto [ceras::normalization_batch](#) (T const momentum=0.98) noexcept
- template<typename T >
requires std::floating_point< T > auto [ceras::batch_normalization](#) (T const momentum=0.98) noexcept
- template<typename T = double>
requires std::floating_point< T > auto [ceras::normalization_instance](#) (T const momentum=0.98) noexcept
- template<typename T >
requires std::floating_point< T > auto [ceras::instance_normalization](#) (T const momentum=0.98) noexcept
- template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto [ceras::concatenate](#) (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept
- auto [ceras::concatenate](#) (unsigned long axe=-1)
- template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto [ceras::concat](#) (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept
- auto [ceras::concat](#) (unsigned long axe=-1)
- template<Expression Lhs_Expression, Expression Rhs_Expression>
constexpr auto [ceras::maximum](#) (Lhs_Expression const &lhs_ex, Rhs_Expression const &rhs_ex) noexcept

Variables

- static constexpr auto [ceras::make_unary_operator](#)
- static constexpr auto [ceras::make_binary_operator](#)
- template<class T >
constexpr bool [ceras::is_operator_v](#) = is_operator<T>::value
- template<typename T >
concept [ceras::Operator](#) = is_operator_v<T>
- template<typename T >
concept [ceras::Expression](#) = Operator<T> || Variable<T> || Place_Holder<T> || Constant<T> || Value<T>

7.10 /data/structured_↵

folders/workspace/github.repo/ceras/include/optimizer.hpp File Reference

```
#include "../config.hpp"
#include "../operation.hpp"
```



```
#include "../place_holder.hpp"
#include "../variable.hpp"
#include "../session.hpp"
#include "../utils/color.hpp"
#include "../utils/debug.hpp"
#include "../utils/id.hpp"
#include "../utils/enable_shared.hpp"
```

Classes

- struct [ceras::sgd< Loss, T >](#)
- struct [ceras::adagrad< Loss, T >](#)
- struct [ceras::rmsprop< Loss, T >](#)
- struct [ceras::adadelta< Loss, T >](#)
- struct [ceras::adam< Loss, T >](#)
- struct [ceras::gradient_descent< Loss, T >](#)

Namespaces

- [ceras](#)

Typedefs

- template<typename Loss , typename T >
using [ceras::ada_grad](#) = adagrad< Loss, T >
- template<typename Loss , typename T >
using [ceras::rms_prop](#) = rmsprop< Loss, T >
- template<typename Loss , typename T >
using [ceras::ada_delta](#) = adadelta< Loss, T >

7.11 /data/structured_folders/workspace/github.repo/ceras/include/place_holder.hpp File Reference

```
#include "../includes.hpp"
#include "../tensor.hpp"
#include "../utils/better_assert.hpp"
#include "../utils/debug.hpp"
#include "../utils/id.hpp"
#include "../utils/enable_shared.hpp"
#include "../utils/state.hpp"
```

Classes

- struct [ceras::place_holder_state< Tsor >](#)
- struct [ceras::place_holder< Tsor >](#)
- struct [ceras::is_place_holder< T >](#)
- struct [ceras::is_place_holder< place_holder< Tsor > >](#)

Namespaces

- [ceras](#)

Variables

- `template<class T >`
`constexpr bool ceras::is_place_holder_v = is_place_holder<T>::value`
- `template<typename T >`
`concept ceras::Place_Holder = is_place_holder_v<T>`

7.12 `/data/structured_↵` **folders/workspace/github.repo/ceras/include/recurrent_↵** **operation.hpp File Reference**

```
#include "../operation.hpp"
#include "../activation.hpp"
#include "../variable.hpp"
```

Namespaces

- [ceras](#)

Functions

- `template<Expression Lhs_Expression, Variable Rhs_Variable>`
`constexpr auto ceras::copy (Lhs_Expression const &lhs_ex, Rhs_Variable const &rhs_va) noexcept`

Variables

- `auto ceras::lstm`

7.13 `/data/structured_↵` **folders/workspace/github.repo/ceras/include/session.hpp File** **Reference**

```
#include "../includes.hpp"
#include "../tensor.hpp"
#include "../place_holder.hpp"
#include "../variable.hpp"
#include "../utils/singleton.hpp"
#include "../utils/debug.hpp"
```

Classes

- struct [ceras::session< T_{src} >](#)

Namespaces

- [ceras](#)

Functions

- template<Tensor T_{src}>
std::reference_wrapper< session< T_{src} > > [ceras::get_default_session](#) ()

7.14 /data/structured_folders/workspace/github.repo/ceras/include/tensor.hpp File Reference

```
#include "../includes.hpp"
#include "../utils/better_assert.hpp"
#include "../utils/range.hpp"
#include "../utils/stride_iterator.hpp"
#include "../utils/for_each.hpp"
#include "../utils/buffered_allocator.hpp"
#include "../utils/debug.hpp"
#include "../utils/id.hpp"
#include "../backend/cuda.hpp"
```

Classes

- struct [ceras::tensor< T, Allocator >](#)
- struct [ceras::is_tensor< T >](#)
- struct [ceras::is_tensor< tensor< T, A > >](#)
- struct [ceras::view_2d< T >](#)
- struct [ceras::view_3d< T >](#)
- struct [ceras::view_4d< T >](#)

Namespaces

- [ceras](#)

Typedefs

- template<typename T >
using [ceras::default_allocator](#) = std::allocator< T >

Functions

- `template<typename T , typename A = default_allocator<T>>>
constexpr tensor< T, A > ceras::as_tensor (T val) noexcept`
- `template<Tensor Tsr, typename CharT , typename Traits >
std::basic_ostream< CharT, Traits > & ceras::operator<< (std::basic_ostream< CharT, Traits > &os_, Tsr
const &tsr)`
- `template<typename T >
requires std::floating_point< T > void ceras::gemm_cpu (T const *A, bool a_transposed, T const *B, bool
b_transposed, unsigned long m, unsigned long n, unsigned long k, T *C)`
- `void ceras::update_cuda_gemm_threshold ()`
- `template<typename T >
requires std::floating_point< T > void ceras::gemm (T const *A, bool a_transposed, T const *B, bool b_
transposed, unsigned long m, unsigned long n, unsigned long k, T *C)`
- `template<typename T >
requires std::floating_point< T > void ceras::gemm (view_2d< T > const &x, view_2d< T > const &y,
view_2d< T > &ans)`
- `template<Tensor Tsr>
Tsr ceras::add (Tsr const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::operator+ (Tsr const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::operator+ (typename Tsr::value_type const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::operator+ (Tsr const &lhs, typename Tsr::value_type const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::minus (Tsr const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::operator- (Tsr const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::operator- (typename Tsr::value_type const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::operator- (Tsr const &lhs, typename Tsr::value_type const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::operator* (typename Tsr::value_type const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::operator* (Tsr const &lhs, typename Tsr::value_type const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::operator/ (Tsr const &lhs, typename Tsr::value_type const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::reshape (Tsr const &ts, std::vector< unsigned long > const &new_shape)`
- `template<Tensor Tsr>
void ceras::multiply (Tsr const &lhs, Tsr const &rhs, Tsr &ans) noexcept`
- `template<Tensor Tsr>
Tsr ceras::multiply (Tsr const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::operator* (Tsr const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::elementwise_product (Tsr const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::hadamard_product (Tsr const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::elementwise_divide (Tsr const &lhs, Tsr const &rhs) noexcept`
- `template<Tensor Tsr>
Tsr ceras::repeat (Tsr const &tsr, unsigned long n)`
- `template<Tensor Tsr>
Tsr ceras::reduce_sum (Tsr const &tsr)`

- `template<Tensor Tsor>`
`Tsor` [ceras::reduce_mean](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`Tsor` [ceras::clip](#) (`Tsor &tsor`, `typename Tsor::value_type lower=0`, `typename Tsor::value_type upper=1`)
- `template<Tensor Tsor>`
`Tsor` [ceras::squeeze](#) (`Tsor const &tsor`)
- `template<typename T, typename A = default_allocator<T>>>`
`tensor< T, A >` [ceras::randn](#) (`std::vector< unsigned long > const &shape`, `T mean=T{0}`, `T stddev=T{1}`)
- `template<typename T, typename A = default_allocator<T>>>`
`tensor< T, A >` [ceras::truncated_normal](#) (`std::vector< unsigned long > const &shape`, `T mean=T{0}`, `T stddev=T{1}`, `T lower=T{0}`, `T upper=T{1}`)
- `template<typename T, typename A = default_allocator<T>>>`
`tensor< T, A >` [ceras::random](#) (`std::vector< unsigned long > const &shape`, `T min=T{0}`, `T max=T{1}`)
- `template<Tensor Tsor>`
`Tsor` [ceras::random_like](#) (`Tsor const &tsor`, `typename Tsor::value_type min=0`, `typename Tsor::value_type max=1`)
- `template<typename T, typename A = default_allocator<T>>>`
`tensor< T, A >` [ceras::glorot_uniform](#) (`std::initializer_list< unsigned long > shape`)
- `template<Tensor Tsor>`
`Tsor` [ceras::deep_copy](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`Tsor` [ceras::copy](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`Tsor` [ceras::concatenate](#) (`Tsor const &lhs`, `Tsor const &rhs`, `unsigned long axis=0`) `noexcept`
- `template<Tensor Tsor>`
`Tsor` [ceras::repmat](#) (`Tsor const &tsor`, `unsigned long row_rep`, `unsigned long col_rep`)
- `template<Tensor Tsor>`
`constexpr bool` [ceras::empty](#) (`Tsor const &tsor`) `noexcept`
- `template<typename T, typename A = default_allocator<T>>>`
`constexpr tensor< T, A >` [ceras::zeros](#) (`std::vector< unsigned long > const &shape`)
- `template<Tensor Tsor>`
`constexpr Tsor` [ceras::zeros_like](#) (`Tsor const &tsor`)
- `template<typename T, typename A = default_allocator<T>>>`
`constexpr tensor< T, A >` [ceras::ones](#) (`std::vector< unsigned long > const &shape`)
- `template<Tensor Tsor>`
`constexpr Tsor` [ceras::ones_like](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`auto` [ceras::max](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`auto` [ceras::amax](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`auto` [ceras::min](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`auto` [ceras::amin](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`auto` [ceras::sum](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`auto` [ceras::mean](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`auto` [ceras::norm](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`Tsor` [ceras::abs](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`Tsor` [ceras::softmax](#) (`Tsor const &tsor`)
- `template<Tensor Tsor>`
`bool` [ceras::has_nan](#) (`Tsor const &tsor`)

- `template<Tensor Tsor>`
`bool ceras::has_inf (Tsor const &tsor)`
- `template<Tensor Tsor>`
`bool ceras::is_valid (Tsor const &tsor)`
- `template<Tensor Tsor, typename Function >`
`Tsor ceras::reduce (Tsor const &ts, unsigned long axis, typename Tsor::value_type const &init, Function const &func, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`Tsor ceras::sum (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`requires std::floating_point< typename Tsor::value_type > Tsor ceras::mean (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`requires std::floating_point< typename Tsor::value_type > Tsor ceras::variance (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`requires std::floating_point< typename Tsor::value_type > Tsor ceras::standard_deviation (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`Tsor ceras::max (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<Tensor Tsor>`
`Tsor ceras::min (Tsor const &ts, unsigned long axis, bool keepdims=false) noexcept`
- `template<typename T, typename A = default_allocator<T>>`
`requires std::floating_point< T > tensor< T, A > ceras::linspace (T start, T stop, unsigned long num, bool endpoint=true) noexcept`
- `template<class _Tp, class _CharT, class _Traits, class _Alloc >`
`std::basic_istream< _CharT, _Traits > & ceras::read_tensor (std::basic_istream< _CharT, _Traits > &__is, tensor< _Tp, _Alloc > &__x)`
- `template<class _Tp, class _CharT, class _Traits, class _Alloc >`
`std::basic_ostream< _CharT, _Traits > & ceras::write_tensor (std::basic_ostream< _CharT, _Traits > &__os, tensor< _Tp, _Alloc > const &__x)`
- `template<typename T, typename A = default_allocator<T>>`
`tensor< T, A > ceras::load_tensor (std::string const &file_name)`
- `template<Tensor Tsor>`
`void ceras::save_tensor (std::string const &file_name, Tsor const &tsor)`

Variables

- static unsigned long `ceras::random_seed` = `std::chrono::system_clock::now().time_since_epoch().count()`
- static `std::mt19937 ceras::random_generator {random_seed}`
- `template<class T >`
`constexpr bool ceras::is_tensor_v = is_tensor<T>::value`
- `template<typename T >`
`concept ceras::Tensor = is_tensor_v<T>`

7.15 /data/structured_↵

folders/workspace/github.repo/ceras/include/value.hpp File Reference

```
#include "../includes.hpp"
#include "../tensor.hpp"
#include "../utils/id.hpp"
#include "../utils/better_assert.hpp"
#include "../utils/enable_shared.hpp"
```

Classes

- struct [ceras::value< T >](#)
- struct [ceras::is_value< T >](#)
- struct [ceras::is_value< value< T > >](#)
- struct [ceras::tensor_deduction< L, R >](#)

Namespaces

- [ceras](#)

Variables

- template<class T >
constexpr bool [ceras::is_value_v](#) = is_value<T>::value
- template<typename T >
concept [ceras::Value](#) = is_value_v<T>

7.16 /data/structured_folders/workspace/github.repo/ceras/include/variable.hpp File Reference

```
#include "../includes.hpp"
#include "../tensor.hpp"
#include "../utils/id.hpp"
#include "../utils/debug.hpp"
#include "../config.hpp"
#include "../utils/enable_shared.hpp"
#include "../utils/state.hpp"
```

Classes

- struct [ceras::variable_state< Tsr >](#)
- struct [ceras::variable< Tsr >](#)
- struct [ceras::is_variable< T >](#)
- struct [ceras::is_variable< variable< Tsr > >](#)

Namespaces

- [ceras](#)

Functions

- template<Tensor Tsr>
std::reference_wrapper< session< Tsr > > [ceras::get_default_session](#) ()
- template<Variable Var>
bool [ceras::operator==](#) (Var const &lhs, Var const &rhs) noexcept

Variables

- template<class T >
constexpr bool [ceras::is_variable_v](#) = is_variable<T>::value
- template<typename T >
concept [ceras::Variable](#) = is_variable_v<T>

Index

/data/structured_folders/workspace/github.repo/ceras/include/activation.hpp, 50
105
/data/structured_folders/workspace/github.repo/ceras/include/ceras.hpp, 17
106
/data/structured_folders/workspace/github.repo/ceras/include/config.hpp, 76
106
/data/structured_folders/workspace/github.repo/ceras/include/constants.hpp, 17
107
/data/structured_folders/workspace/github.repo/ceras/include/inclases.hpp, 17
107
/data/structured_folders/workspace/github.repo/ceras/include/keras.hpp, 50
109
/data/structured_folders/workspace/github.repo/ceras/include/layer.hpp, 17
109
/data/structured_folders/workspace/github.repo/ceras/include/loss.hpp, 17
109
/data/structured_folders/workspace/github.repo/ceras/include/math.hpp, 110
110
/data/structured_folders/workspace/github.repo/ceras/include/optimizer.hpp, 53
112
/data/structured_folders/workspace/github.repo/ceras/include/place_holder.hpp, 64
113
/data/structured_folders/workspace/github.repo/ceras/include/recurrent_operation.hpp, 86
114
/data/structured_folders/workspace/github.repo/ceras/include/session.hpp, 92
114
/data/structured_folders/workspace/github.repo/ceras/include/tensor.hpp, 53
115
/data/structured_folders/workspace/github.repo/ceras/include/value.hpp, 87
118
/data/structured_folders/workspace/github.repo/ceras/include/variable.hpp, 18
119
__version__
 ceras, 39
~session
 ceras::session< Tsor >, 69

abs
 ceras, 16
abs_loss
 ceras, 17
ada_delta
 ceras, 16
ada_grad
 ceras, 16
adadelata
 ceras::adadelata< Loss, T >, 46
adagrad
 ceras::adagrad< Loss, T >, 48
adam
 ceras::adam< Loss, T >, 50
 ceras::adam< Loss, T >, 50
 ceras::binary_operator< Lhs_Operator, Rh_Operator, Forward_Action, Backward_Action >, 53
 ceras::constant< Tsor >, 55
 ceras::place_holder< Tsor >, 64
 ceras::unary_operator< Operator, Forward_Action, Backward_Action >, 86
 ceras::value< T >, 89
 ceras::variable< Tsor >, 92
 backward_action_
 ceras::binary_operator< Lhs_Operator, Rh_Operator, Forward_Action, Backward_Action >, 53
 ceras::unary_operator< Operator, Forward_Action, Backward_Action >, 87
batch_normalization
 ceras, 18
batch_size_
 ceras::view_4d< T >, 103
BatchNormalization
 ceras, 18
begin
 ceras::tensor< T, Allocator >, 78
beta_1_
 ceras::adam< Loss, T >, 50
beta_2_
 ceras::adam< Loss, T >, 51
binary_operator
 ceras::binary_operator< Lhs_Operator, Rh_Operator, Forward_Action, Backward_Action >, 52
bind
 ceras::place_holder< Tsor >, 64
 ceras::session< Tsor >, 70
blas_mode
 ceras, 39

- cbegin
 - ceras::tensor< T, Allocator >, 78
- cend
 - ceras::tensor< T, Allocator >, 78
- ceras, 9
 - __version__, 39
 - abs, 16
 - abs_loss, 17
 - ada_delta, 16
 - ada_grad, 16
 - add, 17
 - amax, 17
 - amin, 17
 - as_tensor, 17
 - average_pooling_2d, 17
 - batch_normalization, 18
 - BatchNormalization, 18
 - blas_mode, 39
 - clip, 18
 - concat, 18
 - concatenate, 19
 - Constant, 39
 - Conv2D, 19
 - conv2d, 19
 - copy, 20
 - cross_entropy, 20
 - cross_entropy_loss, 20
 - cuda_gemm_threshold, 40
 - cuda_mode, 40
 - debug_mode, 40
 - deep_copy, 20
 - default_allocator, 16
 - Dense, 21
 - drop_out, 21
 - elementwise_divide, 21
 - elementwise_product, 21
 - elu, 21
 - empty, 22
 - eps, 40
 - exp, 22
 - exponential, 22
 - Expression, 40
 - flatten, 22
 - gelu, 22
 - gemm, 22, 23
 - gemm_cpu, 23
 - get_default_session, 23
 - glorot_uniform, 23
 - hadamard_product, 23, 24
 - hard_sigmoid, 24
 - has_inf, 24
 - has_nan, 24
 - hinge_loss, 24
 - identity, 24
 - img2col, 25
 - Input, 25
 - instance_normalization, 25
 - is_constant_v, 40
 - is_operator_v, 40
 - is_place_holder_v, 41
 - is_tensor_v, 41
 - is_valid, 25
 - is_value_v, 41
 - is_variable_v, 41
 - is_windows_platform, 41
 - leaky_relu, 25
 - learning_phase, 42
 - linspace, 25
 - load_tensor, 26
 - log, 26
 - lstm, 42
 - mae, 26
 - make_binary_operator, 42
 - make_unary_operator, 42
 - max, 26
 - max_pooling_2d, 27
 - maximum, 27
 - mean, 27
 - mean_absolute_error, 27
 - mean_reduce, 27
 - mean_squared_error, 28
 - mean_squared_logarithmic_error, 28
 - min, 28
 - minus, 28
 - mse, 29
 - multiply, 29
 - negative, 29
 - norm, 29
 - normalization_batch, 29
 - normalization_instance, 30
 - ones, 30
 - ones_like, 30
 - Operator, 42
 - operator<<, 32
 - operator*, 30, 31
 - operator+, 31
 - operator-, 31, 32
 - operator/, 32
 - operator==, 32
 - Place_Holder, 43
 - plus, 33
 - randn, 33
 - random, 33
 - random_generator, 43
 - random_like, 33
 - random_seed, 43
 - read_tensor, 33
 - reduce, 34
 - reduce_mean, 34
 - reduce_sum, 34
 - relu, 34
 - repeat, 35
 - repmat, 35
 - reshape, 35
 - rms_prop, 16
 - save_tensor, 35

- selu, 35
- sigmoid, 36
- softmax, 36
- softplus, 36
- softsign, 36
- square, 36
- squared_loss, 37
- squeeze, 37
- standard_deviation, 37
- sum, 37
- sum_reduce, 37
- tanh, 38
- Tensor, 43
- transpose, 38
- truncated_normal, 38
- up_sampling_2d, 38
- update_cuda_gemm_threshold, 38
- Value, 43
- Variable, 43
- variance, 38
- version, 43
- visible_device, 43
- write_tensor, 39
- zeros, 39
- zeros_like, 39
- ceras::adadelta< Loss, T >, 45
 - adadelta, 46
 - forward, 46
 - iterations_, 46
 - learning_rate_, 46
 - loss_, 46
 - rho_, 47
 - tensor_type, 45
- ceras::adagrad< Loss, T >, 47
 - adagrad, 48
 - decay_, 48
 - forward, 48
 - iterations_, 48
 - learning_rate_, 48
 - loss_, 49
 - tensor_type, 47
- ceras::adam< Loss, T >, 49
 - adam, 50
 - amsgrad_, 50
 - beta_1_, 50
 - beta_2_, 51
 - forward, 50
 - iterations_, 51
 - learning_rate_, 51
 - loss_, 51
 - tensor_type, 50
- ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >, 51
 - backward, 53
 - backward_action_, 53
 - binary_operator, 52
 - forward, 53
 - forward_action_, 53
 - lhs_input_data_, 53
 - lhs_op_, 54
 - output_data_, 54
 - reset_action_, 54
 - reset_states, 53
 - rhs_input_data_, 54
 - rhs_op_, 54
 - tensor_type, 52
- ceras::constant< Tzor >, 55
 - backward, 55
 - data_, 56
 - forward, 55
 - shape, 55
- ceras::gradient_descent< Loss, T >, 56
 - forward, 57
 - gradient_descent, 57
 - learning_rate_, 57
 - loss_, 57
 - momentum_, 58
 - tensor_type, 57
- ceras::is_constant< constant< Tzor > >, 58
- ceras::is_constant< T >, 58
- ceras::is_operator< binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action > >, 59
- ceras::is_operator< T >, 59
- ceras::is_operator< unary_operator< Operator, Forward_Action, Backward_Action > >, 59
- ceras::is_place_holder< place_holder< Tzor > >, 60
- ceras::is_place_holder< T >, 60
- ceras::is_tensor< T >, 60
- ceras::is_tensor< tensor< T, A > >, 61
- ceras::is_value< T >, 61
- ceras::is_value< value< T > >, 61
- ceras::is_variable< T >, 62
- ceras::is_variable< variable< Tzor > >, 62
- ceras::place_holder< Tzor >, 62
 - backward, 64
 - bind, 64
 - forward, 64
 - operator=, 64
 - place_holder, 63
 - reset, 64
 - tensor_type, 63
- ceras::place_holder_state< Tzor >, 65
 - data_, 65
 - shape_hint_, 65
- ceras::rmsprop< Loss, T >, 65
 - decay_, 67
 - forward, 66
 - iterations_, 67
 - learning_rate_, 67
 - loss_, 67
 - rho_, 67
 - rmsprop, 66
 - tensor_type, 66
- ceras::session< Tzor >, 68
 - ~session, 69

- bind, 70
- deserialize, 70
- operator=, 70
- place_holder_type, 68
- place_holders_, 72
- rebind, 70
- remember, 70
- restore, 71
- run, 71
- save, 71
- serialize, 71
- session, 69
- tap, 71
- variable_state_type, 68
- variable_type, 69
- variables_, 72
- ceras::sgd< Loss, T >, 72
 - decay_, 73
 - forward, 73
 - iterations_, 74
 - learning_rate_, 74
 - loss_, 74
 - momentum_, 74
 - nesterov_, 74
 - sgd, 73
 - tensor_type, 73
- ceras::tensor< T, Allocator >, 75
 - allocator, 76
 - begin, 78
 - cbegin, 78
 - cend, 78
 - copy, 79
 - creep_to, 79
 - data, 79
 - deep_copy, 79
 - empty, 80
 - end, 80
 - map, 80
 - memory_offset_, 84
 - ndim, 80
 - operator*=, 80, 81
 - operator+=, 81
 - operator-, 81
 - operator=, 81
 - operator/=: 82
 - operator=, 82
 - operator[], 82
 - reset, 83
 - reshape, 83
 - resize, 83
 - self_type, 76
 - shape, 83
 - shape_, 84
 - shared_vector, 76
 - shrink_to, 83
 - size, 83
 - slice, 84
 - tensor, 77, 78
 - value_type, 76
 - vector_, 84
 - vector_type, 77
- ceras::tensor_deduction< L, R >, 84
 - op_type, 85
 - tensor_type, 85
- ceras::unary_operator< Operator, Forward_Action, Backward_Action >, 85
 - backward, 86
 - backward_action_, 87
 - forward, 86
 - forward_action_, 87
 - input_data_, 87
 - op_, 87
 - output_data_, 87
 - reset_action_, 87
 - reset_states, 86
 - tensor_type, 87
 - unary_operator, 86
- ceras::value< T >, 88
 - backward, 89
 - data_, 90
 - forward, 89
 - operator=, 90
 - value, 89
 - value_type, 88
- ceras::variable< Tstor >, 90
 - backward, 92
 - data, 92
 - forward, 92
 - gradient, 93
 - operator=, 93
 - reset, 93
 - reset_states, 93
 - shape, 93
 - state_, 94
 - stateful_, 94
 - tensor_type, 91
 - trainable_, 94
 - variable, 91, 92
- ceras::variable_state< Tstor >, 94
 - data_, 94
 - gradient_, 95
- ceras::view_2d< T >, 95
 - col_, 97
 - data, 96
 - data_, 97
 - operator[], 96, 97
 - row_, 97
 - shape, 97
 - size, 97
 - transposed_, 98
 - view_2d, 95, 96
- ceras::view_3d< T >, 98
 - channel_, 99
 - col_, 99
 - data_, 99
 - operator[], 99

- row_, 99
- view_3d, 98
- ceras::view_4d< T >, 100
 - batch_size_, 103
 - channel_, 103
 - col_, 103
 - data_, 103
 - operator[], 102
 - row_, 103
 - view_4d, 100
- channel_
 - ceras::view_3d< T >, 99
 - ceras::view_4d< T >, 103
- clip
 - ceras, 18
- col_
 - ceras::view_2d< T >, 97
 - ceras::view_3d< T >, 99
 - ceras::view_4d< T >, 103
- concat
 - ceras, 18
- concatenate
 - ceras, 19
- Constant
 - ceras, 39
- Conv2D
 - ceras, 19
- conv2d
 - ceras, 19
- copy
 - ceras, 20
 - ceras::tensor< T, Allocator >, 79
- creep_to
 - ceras::tensor< T, Allocator >, 79
- cross_entropy
 - ceras, 20
- cross_entropy_loss
 - ceras, 20
- cuda_gemm_threshold
 - ceras, 40
- cuda_mode
 - ceras, 40
- data
 - ceras::tensor< T, Allocator >, 79
 - ceras::variable< T >, 92
 - ceras::view_2d< T >, 96
- data_
 - ceras::constant< T >, 56
 - ceras::place_holder_state< T >, 65
 - ceras::value< T >, 90
 - ceras::variable_state< T >, 94
 - ceras::view_2d< T >, 97
 - ceras::view_3d< T >, 99
 - ceras::view_4d< T >, 103
- debug_mode
 - ceras, 40
- decay_
 - ceras::adagrad< Loss, T >, 48
 - ceras::rmsprop< Loss, T >, 67
 - ceras::sgd< Loss, T >, 73
- deep_copy
 - ceras, 20
 - ceras::tensor< T, Allocator >, 79
- default_allocator
 - ceras, 16
- Dense
 - ceras, 21
- deserialize
 - ceras::session< T >, 70
- drop_out
 - ceras, 21
- elementwise_divide
 - ceras, 21
- elementwise_product
 - ceras, 21
- elu
 - ceras, 21
- empty
 - ceras, 22
 - ceras::tensor< T, Allocator >, 80
- end
 - ceras::tensor< T, Allocator >, 80
- eps
 - ceras, 40
- exp
 - ceras, 22
- exponential
 - ceras, 22
- Expression
 - ceras, 40
- flatten
 - ceras, 22
- forward
 - ceras::adadelta< Loss, T >, 46
 - ceras::adagrad< Loss, T >, 48
 - ceras::adam< Loss, T >, 50
 - ceras::binary_operator< Lhs_Operator, Rh_Operator, Forward_Action, Backward_Action >, 53
 - ceras::constant< T >, 55
 - ceras::gradient_descent< Loss, T >, 57
 - ceras::place_holder< T >, 64
 - ceras::rmsprop< Loss, T >, 66
 - ceras::sgd< Loss, T >, 73
 - ceras::unary_operator< Operator, Forward_Action, Backward_Action >, 86
 - ceras::value< T >, 89
 - ceras::variable< T >, 92
- forward_action_
 - ceras::binary_operator< Lhs_Operator, Rh_Operator, Forward_Action, Backward_Action >, 53
 - ceras::unary_operator< Operator, Forward_Action, Backward_Action >, 87
- gelu
 - ceras, 22

- gemm
 - ceras, [22](#), [23](#)
- gemm_cpu
 - ceras, [23](#)
- get_default_session
 - ceras, [23](#)
- glorot_uniform
 - ceras, [23](#)
- gradient
 - ceras::variable< T_{sor} >, [93](#)
- gradient_
 - ceras::variable_state< T_{sor} >, [95](#)
- gradient_descent
 - ceras::gradient_descent< Loss, T >, [57](#)
- hadamard_product
 - ceras, [23](#), [24](#)
- hard_sigmoid
 - ceras, [24](#)
- has_inf
 - ceras, [24](#)
- has_nan
 - ceras, [24](#)
- hinge_loss
 - ceras, [24](#)
- identity
 - ceras, [24](#)
- img2col
 - ceras, [25](#)
- includes.hpp
 - STB_IMAGE_IMPLEMENTATION, [108](#)
 - STB_IMAGE_RESIZE_IMPLEMENTATION, [108](#)
 - STB_IMAGE_WRITE_IMPLEMENTATION, [108](#)
- Input
 - ceras, [25](#)
- input_data_
 - ceras::unary_operator< Operator, Forward_Action, Backward_Action >, [87](#)
- instance_normalization
 - ceras, [25](#)
- is_constant_v
 - ceras, [40](#)
- is_operator_v
 - ceras, [40](#)
- is_place_holder_v
 - ceras, [41](#)
- is_tensor_v
 - ceras, [41](#)
- is_valid
 - ceras, [25](#)
- is_value_v
 - ceras, [41](#)
- is_variable_v
 - ceras, [41](#)
- is_windows_platform
 - ceras, [41](#)
- iterations_
 - ceras::adadelta< Loss, T >, [46](#)
 - ceras::adagrad< Loss, T >, [48](#)
 - ceras::adam< Loss, T >, [51](#)
 - ceras::rmsprop< Loss, T >, [67](#)
 - ceras::sgd< Loss, T >, [74](#)
- leaky_relu
 - ceras, [25](#)
- learning_phase
 - ceras, [42](#)
- learning_rate_
 - ceras::adadelta< Loss, T >, [46](#)
 - ceras::adagrad< Loss, T >, [48](#)
 - ceras::adam< Loss, T >, [51](#)
 - ceras::gradient_descent< Loss, T >, [57](#)
 - ceras::rmsprop< Loss, T >, [67](#)
 - ceras::sgd< Loss, T >, [74](#)
- lhs_input_data_
 - ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >, [53](#)
- lhs_op_
 - ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >, [54](#)
- linspace
 - ceras, [25](#)
- load_tensor
 - ceras, [26](#)
- log
 - ceras, [26](#)
- loss_
 - ceras::adadelta< Loss, T >, [46](#)
 - ceras::adagrad< Loss, T >, [49](#)
 - ceras::adam< Loss, T >, [51](#)
 - ceras::gradient_descent< Loss, T >, [57](#)
 - ceras::rmsprop< Loss, T >, [67](#)
 - ceras::sgd< Loss, T >, [74](#)
- lstm
 - ceras, [42](#)
- mae
 - ceras, [26](#)
- make_binary_operator
 - ceras, [42](#)
- make_unary_operator
 - ceras, [42](#)
- map
 - ceras::tensor< T, Allocator >, [80](#)
- max
 - ceras, [26](#)
- max_pooling_2d
 - ceras, [27](#)
- maximum
 - ceras, [27](#)
- mean
 - ceras, [27](#)
- mean_absolute_error
 - ceras, [27](#)
- mean_reduce
 - ceras, [27](#)
- mean_squared_error

- ceras, [28](#)
- mean_squared_logarithmic_error
 - ceras, [28](#)
- memory_offset_
 - ceras::tensor< T, Allocator >, [84](#)
- min
 - ceras, [28](#)
- minus
 - ceras, [28](#)
- momentum_
 - ceras::gradient_descent< Loss, T >, [58](#)
 - ceras::sgd< Loss, T >, [74](#)
- mse
 - ceras, [29](#)
- multiply
 - ceras, [29](#)
- ndim
 - ceras::tensor< T, Allocator >, [80](#)
- negative
 - ceras, [29](#)
- nesterov_
 - ceras::sgd< Loss, T >, [74](#)
- norm
 - ceras, [29](#)
- normalization_batch
 - ceras, [29](#)
- normalization_instance
 - ceras, [30](#)
- ones
 - ceras, [30](#)
- ones_like
 - ceras, [30](#)
- op_
 - ceras::unary_operator< Operator, Forward_Action, Backward_Action >, [87](#)
- op_type
 - ceras::tensor_deduction< L, R >, [85](#)
- Operator
 - ceras, [42](#)
- operator<<
 - ceras, [32](#)
- operator*
 - ceras, [30](#), [31](#)
- operator*=
 - ceras::tensor< T, Allocator >, [80](#), [81](#)
- operator+
 - ceras, [31](#)
- operator+=
 - ceras::tensor< T, Allocator >, [81](#)
- operator-
 - ceras, [31](#), [32](#)
 - ceras::tensor< T, Allocator >, [81](#)
- operator-=
 - ceras::tensor< T, Allocator >, [81](#)
- operator/
 - ceras, [32](#)
- operator/=
 - ceras::tensor< T, Allocator >, [82](#)
- operator=
 - ceras::place_holder< T, Allocator >, [64](#)
 - ceras::session< T, Allocator >, [70](#)
 - ceras::tensor< T, Allocator >, [82](#)
 - ceras::value< T >, [90](#)
 - ceras::variable< T, Allocator >, [93](#)
- operator[]
 - ceras::tensor< T, Allocator >, [82](#)
 - ceras::view_2d< T >, [96](#), [97](#)
 - ceras::view_3d< T >, [99](#)
 - ceras::view_4d< T >, [102](#)
- output_data_
 - ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >, [54](#)
 - ceras::unary_operator< Operator, Forward_Action, Backward_Action >, [87](#)
- Place_Holder
 - ceras, [43](#)
- place_holder
 - ceras::place_holder< T, Allocator >, [63](#)
- place_holder_type
 - ceras::session< T, Allocator >, [68](#)
- place_holders_
 - ceras::session< T, Allocator >, [72](#)
- plus
 - ceras, [33](#)
- randn
 - ceras, [33](#)
- random
 - ceras, [33](#)
- random_generator
 - ceras, [43](#)
- random_like
 - ceras, [33](#)
- random_seed
 - ceras, [43](#)
- read_tensor
 - ceras, [33](#)
- rebind
 - ceras::session< T, Allocator >, [70](#)
- reduce
 - ceras, [34](#)
- reduce_mean
 - ceras, [34](#)
- reduce_sum
 - ceras, [34](#)
- relu
 - ceras, [34](#)
- remember
 - ceras::session< T, Allocator >, [70](#)
- repeat
 - ceras, [35](#)
- repmat
 - ceras, [35](#)

reset
 ceras::place_holder< T, Allocator >, 64
 ceras::tensor< T, Allocator >, 83
 ceras::variable< T, Allocator >, 93
 reset_action_
 ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >, 54
 ceras::unary_operator< Operator, Forward_Action, Backward_Action >, 87
 reset_states
 ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >, 53
 ceras::unary_operator< Operator, Forward_Action, Backward_Action >, 86
 ceras::variable< T, Allocator >, 93
 reshape
 ceras, 35
 ceras::tensor< T, Allocator >, 83
 resize
 ceras::tensor< T, Allocator >, 83
 restore
 ceras::session< T, Allocator >, 71
 rho_
 ceras::adadelat< Loss, T >, 47
 ceras::rmsprop< Loss, T >, 67
 rhs_input_data_
 ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >, 54
 rhs_op_
 ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >, 54
 rms_prop
 ceras, 16
 rmsprop
 ceras::rmsprop< Loss, T >, 66
 row_
 ceras::view_2d< T >, 97
 ceras::view_3d< T >, 99
 ceras::view_4d< T >, 103
 run
 ceras::session< T, Allocator >, 71
 save
 ceras::session< T, Allocator >, 71
 save_tensor
 ceras, 35
 self_type
 ceras::tensor< T, Allocator >, 76
 selu
 ceras, 35
 serialize
 ceras::session< T, Allocator >, 71
 session
 ceras::session< T, Allocator >, 69
 sgd
 ceras::sgd< Loss, T >, 73
 shape
 ceras::constant< T, Allocator >, 55
 ceras::tensor< T, Allocator >, 83
 ceras::variable< T, Allocator >, 93
 ceras::view_2d< T >, 97
 shape_
 ceras::tensor< T, Allocator >, 84
 shape_hint_
 ceras::place_holder_state< T, Allocator >, 65
 shared_vector
 ceras::tensor< T, Allocator >, 76
 shrink_to
 ceras::tensor< T, Allocator >, 83
 sigmoid
 ceras, 36
 size
 ceras::tensor< T, Allocator >, 83
 ceras::view_2d< T >, 97
 slice
 ceras::tensor< T, Allocator >, 84
 softmax
 ceras, 36
 softplus
 ceras, 36
 softsign
 ceras, 36
 square
 ceras, 36
 squared_loss
 ceras, 37
 squeeze
 ceras, 37
 standard_deviation
 ceras, 37
 state_
 ceras::variable< T, Allocator >, 94
 stateful_
 ceras::variable< T, Allocator >, 94
 STB_IMAGE_IMPLEMENTATION
 includes.hpp, 108
 STB_IMAGE_RESIZE_IMPLEMENTATION
 includes.hpp, 108
 STB_IMAGE_WRITE_IMPLEMENTATION
 includes.hpp, 108
 sum
 ceras, 37
 sum_reduce
 ceras, 37
 tanh
 ceras, 38
 tap
 ceras::session< T, Allocator >, 71
 Tensor
 ceras, 43
 tensor
 ceras::tensor< T, Allocator >, 77, 78
 tensor_type
 ceras::adadelat< Loss, T >, 45
 ceras::adagrad< Loss, T >, 47
 ceras::adam< Loss, T >, 50

- ceras::binary_operator< Lhs_Operator, Rhs_Operator, Forward_Action, Backward_Action >, 52
 - ceras::gradient_descent< Loss, T >, 57
 - ceras::place_holder< Tsr >, 63
 - ceras::rmsprop< Loss, T >, 66
 - ceras::sgd< Loss, T >, 73
 - ceras::tensor_deduction< L, R >, 85
 - ceras::unary_operator< Operator, Forward_Action, Backward_Action >, 87
 - ceras::variable< Tsr >, 91
- trainable_
 - ceras::variable< Tsr >, 94
- transpose
 - ceras, 38
- transposed_
 - ceras::view_2d< T >, 98
- truncated_normal
 - ceras, 38
- unary_operator
 - ceras::unary_operator< Operator, Forward_Action, Backward_Action >, 86
- up_sampling_2d
 - ceras, 38
- update_cuda_gemm_threshold
 - ceras, 38
- Value
 - ceras, 43
- value
 - ceras::value< T >, 89
- value_type
 - ceras::tensor< T, Allocator >, 76
 - ceras::value< T >, 88
- Variable
 - ceras, 43
- variable
 - ceras::variable< Tsr >, 91, 92
- variable_state_type
 - ceras::session< Tsr >, 68
- variable_type
 - ceras::session< Tsr >, 69
- variables_
 - ceras::session< Tsr >, 72
- variance
 - ceras, 38
- vector_
 - ceras::tensor< T, Allocator >, 84
- vector_type
 - ceras::tensor< T, Allocator >, 77
- version
 - ceras, 43
- view_2d
 - ceras::view_2d< T >, 95, 96
- view_3d
 - ceras::view_3d< T >, 98
- view_4d
 - ceras::view_4d< T >, 100
- visible_device