Module: tf.keras.backend

Defined in `tensorflow/python/keras/backend/__init__.py`.

Keras backend API.

Classes

`class name_scope`: A context manager for use when defining a Python op.

Functions

`abs(...)`: Element-wise absolute value.

`all(...)`: Bitwise reduction (logical AND).

`any(...)`: Bitwise reduction (logical OR).

`arange(...)`: Creates a 1D tensor containing a sequence of integers.

`argmax(...)`: Returns the index of the maximum value along an axis.

`argmin(...)`: Returns the index of the minimum value along an axis.

`backend(...)`: Publicly accessible method for determining the current backend.

`batch_dot(...)`: Batchwise dot product.

`batch_flatten(...)`: Turn a nD tensor into a 2D tensor with same 0th dimension.

`batch_get_value(...)`: Returns the value of more than one tensor variable.

`batch_normalization(...)`: Applies batch normalization on x given mean, var, beta and gamma.

`batch_set_value(...)`: Sets the values of many tensor variables at once.

`bias_add(...)`: Adds a bias vector to a tensor.

`binary_crossentropy(...)`: Binary crossentropy between an output tensor and a target tensor.

`cast(...)`: Casts a tensor to a different dtype and returns it.

`cast_to_floatx(...)`: Cast a Numpy array to the default Keras float type.

`categorical_crossentropy(...)`: Categorical crossentropy between an output tensor and a target tensor.

`clear_session(...)`: Destroys the current TF graph and creates a new one.

`clip(...)`: Element-wise value clipping.

`concatenate(...)` : Concatenates a list of tensors alongside the specified axis.

`constant(...)` : Creates a constant tensor.

`conv1d(...)` : 1D convolution.

`conv2d(...)` : 2D convolution.

`conv2d_transpose(...)` : 2D deconvolution (i.e.

`conv3d(...)` : 3D convolution.

`cos(...)` : Computes cos of x element-wise.

`count_params(...)` : Returns the number of scalars in a Keras variable.

`ctc_batch_cost(...)` : Runs CTC loss algorithm on each batch element.

`ctc_decode(...)` : Decodes the output of a softmax.

`ctc_label_dense_to_sparse(...)` : Converts CTC labels from dense to sparse.

`dot(...)` : Multiplies 2 tensors (and/or variables) and returns a *tensor*.

`dropout(...)` : Sets entries in `x` to zero at random, while scaling the entire tensor.

`dtype(...)` : Returns the dtype of a Keras tensor or variable, as a string.

`elu(...)` : Exponential linear unit.

`epsilon(...)` : Returns the value of the fuzz factor used in numeric expressions.

`equal(...)` : Element-wise equality between two tensors.

`eval(...)` : Evaluates the value of a variable.

`exp(...)` : Element-wise exponential.

`expand_dims(...)` : Adds a 1-sized dimension at index "axis".

`eye(...)` : Instantiate an identity matrix and returns it.

`flatten(...)` : Flatten a tensor.

`floatx(...)` : Returns the default float type, as a string.

`foldl(...)` : Reduce elems using fn to combine them from left to right.

`foldr(...)` : Reduce elems using fn to combine them from right to left.

`function(...)` : Instantiates a Keras function.

`gather(...)` : Retrieves the elements of indices `indices` in the tensor `reference`.

`get_session(...)` : Returns the TF session to be used by the backend.

`get_uid(...)` : Associates a string prefix with an integer counter in a TensorFlow graph.

`get_value(...)` : Returns the value of a variable.

`gradients(...)` : Returns the gradients of `variables` w.r.t. `loss`.

`greater(...)` : Element-wise truth value of (x > y).

`greater_equal(...)` : Element-wise truth value of (x >= y).

`hard_sigmoid(...)` : Segment-wise linear approximation of sigmoid.

`image_data_format(...)` : Returns the default image data format convention.

`in_test_phase(...)` : Selects `x` in test phase, and `alt` otherwise.

`in_top_k(...)` : Returns whether the `targets` are in the top `k` `predictions`.

`in_train_phase(...)` : Selects `x` in train phase, and `alt` otherwise.

`int_shape(...)` : Returns the shape tensor or variable as a tuple of int or None entries.

`is_sparse(...)` : Returns whether a tensor is a sparse tensor.

`l2_normalize(...)` : Normalizes a tensor wrt the L2 norm alongside the specified axis.

`learning_phase(...)` : Returns the learning phase flag.

`less(...)` : Element-wise truth value of (x < y).

`less_equal(...)` : Element-wise truth value of (x <= y).

`log(...)` : Element-wise log.

`manual_variable_initialization(...)` : Sets the manual variable initialization flag.

`map_fn(...)` : Map the function fn over the elements elems and return the outputs.

`max(...)` : Maximum value in a tensor.

`maximum(...)` : Element-wise maximum of two tensors.

`mean(...)` : Mean of a tensor, alongside the specified axis.

`min(...)` : Minimum value in a tensor.

`minimum(...)` : Element-wise minimum of two tensors.

`moving_average_update(...)` : Compute the moving average of a variable.

`ndim(...)` : Returns the number of axes in a tensor, as an integer.

`normalize_batch_in_training(...)` : Computes mean and std for batch then apply batch_normalization on batch.

`not_equal(...)` : Element-wise inequality between two tensors.

`one_hot(...)` : Computes the one-hot representation of an integer tensor.

`ones(...)` : Instantiates an all-ones tensor variable and returns it.

`ones_like(...)` : Instantiates an all-ones variable of the same shape as another tensor.

`permute_dimensions(...)` : Permutes axes in a tensor.

`placeholder(...)` : Instantiates a placeholder tensor and returns it.

`pool2d(...)` : 2D Pooling.

`pool3d(...)` : 3D Pooling.

`pow(...)` : Element-wise exponentiation.

`print_tensor(...)` : Prints `message` and the tensor value when evaluated.

`prod(...)` : Multiplies the values in a tensor, alongside the specified axis.

`random_binomial(...)` : Returns a tensor with random binomial distribution of values.

`random_normal(...)` : Returns a tensor with normal distribution of values.

`random_normal_variable(...)` : Instantiates a variable with values drawn from a normal distribution.

`random_uniform(...)` : Returns a tensor with uniform distribution of values.

`random_uniform_variable(...)` : Instantiates a variable with values drawn from a uniform distribution.

`relu(...)` : Rectified linear unit.

`repeat(...)` : Repeats a 2D tensor.

`repeat_elements(...)` : Repeats the elements of a tensor along an axis, like `np.repeat`.

`reset_uids(...)`

`reshape(...)` : Reshapes a tensor to the specified shape.

`resize_images(...)` : Resizes the images contained in a 4D tensor.

`resize_volumes(...)` : Resizes the volume contained in a 5D tensor.

`reverse(...)` : Reverse a tensor along the specified axes.

`rnn(...)` : Iterates over the time dimension of a tensor.

`round(...)` : Element-wise rounding to the closest integer.

`separable_conv2d(...)` : 2D convolution with separable filters.

`set_epsilon(...)` : Sets the value of the fuzz factor used in numeric expressions.

`set_floatx(...)` : Sets the default float type.

`set_image_data_format(...)` : Sets the value of the image data format convention.

`set_learning_phase(...)` : Sets the learning phase to a fixed value.

`set_session(...)` : Sets the global TensorFlow session.

`set_value(...)` : Sets the value of a variable, from a Numpy array.

`shape(...)` : Returns the symbolic shape of a tensor or variable.

`sigmoid(...)` : Element-wise sigmoid.

`sign(...)` : Element-wise sign.

`sin(...)` : Computes sin of x element-wise.

`softmax(...)` : Softmax of a tensor.

`softplus(...)` : Softplus of a tensor.

`softsign(...)` : Softsign of a tensor.

`sparse_categorical_crossentropy(...)` : Categorical crossentropy with integer targets.

`spatial_2d_padding(...)` : Pads the 2nd and 3rd dimensions of a 4D tensor.

`spatial_3d_padding(...)` : Pads 5D tensor with zeros along the depth, height, width dimensions.

`sqrt(...)` : Element-wise square root.

`square(...)` : Element-wise square.

`squeeze(...)` : Removes a 1-dimension from the tensor at index "axis".

`stack(...)` : Stacks a list of rank `R` tensors into a rank `R+1` tensor.

`std(...)` : Standard deviation of a tensor, alongside the specified axis.

`stop_gradient(...)` : Returns `variables` but with zero gradient w.r.t. every other variable.

`sum(...)` : Sum of the values in a tensor, alongside the specified axis.

`switch(...)` : Switches between two operations depending on a scalar value.

`tanh(...)` : Element-wise tanh.

`temporal_padding(...)` : Pads the middle dimension of a 3D tensor.

`to_dense(...)` : Converts a sparse tensor into a dense tensor and returns it.

`transpose(...)` : Transposes a tensor and returns it.

`truncated_normal(...)` : Returns a tensor with truncated random normal distribution of values.

`update(...)`

`update_add(...)` : Update the value of `x` by adding `increment`.

`update_sub(...)` : Update the value of `x` by subtracting `decrement`.

`var(...)` : Variance of a tensor, alongside the specified axis.

`variable(...)` : Instantiates a variable and returns it.

`zeros(...)` : Instantiates an all-zeros variable and returns it.

`zeros_like(...)` : Instantiates an all-zeros variable of the same shape as another tensor.

Last updated November 2, 2017.