芯驰

SemiDrive

X9 AI OP Spec

Table of Contents

Document Revision

Date	Revision	Description
2022-12-01	Rev 01.00	First draft.

1 Introduction

This document lists all supported AI operators on Semidrive X9 platforms and describes their specifications.

AI computing units on Semidrive X9 platforms include AIPU and SlimAI. The map between AIPU operators and frontend framework operators (Tensorflow, TensorFlow-Lite, ONNX, and Caffe) is listed in Section 2. Restrictions of AIPU operators are listed in Section 3. The operators supported by SimAI are listed in Section 4.

2 AIPU OP Map

The following table shows a map between AIPU operators (IR) and the operators (API) of TensorFlow, TensorFlow-Lite, ONNX, and Caffe.

IR	TensorFlow	TensorFlow-Lite	ONNX	Caffe
Abs	tf.math.abs	tfl.abs	Abs	AbsVal
AccidentalHits	tf.nn.compute_accidental_hits	-	-	-
Acos	tf.math.acos	-	Acos	-
Acosh	tf.math.acosh	-	Acosh	-
Add	tf.math.add_n	tfl.add_n	Sum	Eltwise
ArgMax	tf.math.argmax	tfl.arg_max	ArgMax	ArgMax
ArgMin	tf.math.argmin	tfl.arg_min	ArgMin	-
Asin	tf.math.asin	-	Asin	-
Asinh	tf.math.asinh	-	Asinh	-
AveragePooling2D	tf.nn.avg_pool2d	tfl.average_pool_2d	AveragePool	Pooling
			GlobalAveragePool	
AveragePooling3D	tf.nn.avg_pool3d	-	AveragePool	-
BasicLSTM	tf.keras.layers.LSTM	tfl.unidirectional_sequence_lstm	LSTM	LSTM
BatchNormalization	tf.nn.batch_normalization	-	BatchNormalization	BatchNorm
BatchToSpace	tf.batch_to_space	tfl.batch_to_space_nd	-	-
BiasAdd	tf.nn.bias_add	-	-	Bias
BNLL	-	-	-	BNLL
Cast	tf.cast	tfl.cast	Cast	-
Ceil	tf.math.ceil	tfl.ceil	Ceil	-
Celu	-	-	Celu	-
ChannelShuffle	Reshape+Transpose+Reshape+Split	Reshape+Transpose+Reshape+Split	Reshape + Transpose + Reshape + Split	-
Clip	tf.clip_by_value	-	Clip	-
Compress	-	-	Compress	-
Concat	tf.concat	tfl.concatenation	Concat	Concat
Constant	tf.ones	-	-	-
	tf.ones_like	-	-	-

IR	TensorFlow	TensorFlow-Lite	ONNX	Caffe
	tf.zeros	–	–	–
	tf.zeros_like	–	–	–
	tf.range	tfl.range	Range	–
	tf.constant	–	Constant	–
Convolution3D	tf.nn.conv3d	–	Conv	Convolution
Convolution2D	tf.nn.conv2d	tfl.conv_2d		
ConvTranspose2D	tf.nn.conv2d_transpose	tfl.transpose_conv	ConvTranspose	Deconvolution
ConvTranspose3D	tf.nn.conv3d_transpose	–		
Cosh	tf.math.cosh	–	Cosh	–
Cosine	tf.math.cos	tfl.cos	Cos	–
Count	–	–	–	–
Crelu	tf.nn.crelu	–	–	–
Crop	–	–	–	Crop
CropAndResize	tf.image.crop_and_resize	–	Resize	–
CTCGreedyDecoder	tf.nn.ctc_greedy_decoder	–	–	–
DataStride	–	–	–	–
DepthToSpace	tf.nn.depth_to_space	tfl.depth_to_space	DepthToSpace	–
DepthwiseConvolution	tf.nn.depthwise_conv2d	tfl.depthwise_conv_2d	–	–
Div	tf.math.divide	tfl.div	Div	–
ElementwiseAdd	tf.math.add	tfl.add	Add	Eltwise
ElementwiseMax	tf.math.maximum	tfl.maximum	Max	–
ElementwiseMin	tf.math.minimum	tfl.minimum	Min	–
ElementwiseMul	tf.math.multiply	tfl.mul	Mul	Eltwise
ElementwiseSub	tf.math.subtract	tfl.sub	Sub	–
Elu	tf.nn.elu	tfl.elu	Elu	ELU
Exp	tf.math.exp	tfl.exp	Exp	Exp
Filter	–	–	–	Filter
Floor	tf.math.floor	tfl.floor	Floor	–
FullyConnected	tf.linalg.matmul	tfl.fully_connected	–	InnerProduct
Gather	tf.gather	tfl.gather	Gather	–
GatherElements	–	–	GatherElements	–
GatherND	tf.gather_nd	tfl.gather_nd	GatherND	–

IR	TensorFlow	TensorFlow-Lite	ONNX	Caffe
Gemm	–	–	Gemm	–
GridSample	–	–	GridSample	–
GroupConvolution	split+conv2d+concat	split+conv2d+concat	Conv	Convolution
GroupNorm	tfa.layers.GroupNormalization	–	–	–
GRUv1	tf.keras.layers.GRU	–	GRU	–
GRUv3	–	–		–
HardSigmoid	tf.keras.activations.hard_sigmoid	–	HardSigmoid	–
HardSwish	–	tfl.hard_swish	HardSwish	–
InstanceNormalization	tfa.layers.InstanceNormalization	–	InstanceNormalization	–
InTopK	tf.math.in_top_k	–	–	–
LayerNormalization	tf.keras.layers.LayerNormalization	–	LayerNormalizatin	–
LeakyRelu	tf.nn.leaky_relu	tfl.leaky_relu	LeakyRelu	ReLU
LeftShift	tf.bitwise.left_shift	–	BitShift	–
L1Pooling2D	–	–	LpPool	–
L2Pooling2D	–	–	LpPool	–
Log	tf.math.log	tfl.log	Log	Log
Logical	tf.math.logical_and	tfl.logical_and	And	–
	tf.math.logical_not	tfl.logical_not	Not	–
	tf.math.logical_or	tfl.logical_or	Or	–
	tf.math.logical_xor	–	Xor	–
	tf.math.equal	tfl.equal	Equal	–
	tf.math.not_equal	tfl.not_equal	–	–
	tf.math.greater	tfl.greater	Greater	–
	tf.math.greater_equal	tfl.greater_equal	GreaterOrEqual	–
	tf.math.less	tfl.less	Less	–
	tf.math.less_equal	tfl.less_equal	LessOrEqual	–
LogSoftmax	tf.nn.log_softmax	tfl.log_softmax	LogSoftmax	–
LRN	tf.nn.local_response_normalization	tfl.local_response_normalization	LRN	LRN
MatMul	tf.linalg.matmul	tfl.batch_matmul	MatMul	InnerProduct
MaxPooling2D	tf.nn.max_pool2d	tfl.max_pool_2d	MaxPool	Pooling
			GlobalMaxPool	

IR	TensorFlow	TensorFlow-Lite	ONNX	Caffe
MaxPooling3D	tf.nn.max_pool3d	-	MaxPool	-
MaxPoolingWithArgMax	tf.nn.max_pool_with_argmax	-		Pooling
MaxRoiPool	-	-	MaxRoiPool	-
MaxUnpool	-	-	MaxUnpool	-
MeanVarianceNormalization	-	-	MeanVarianceNormalization	MVN
Mish	tfa.activations.mish	-	-	-
Mod	tf.math.floormod	tfl.floor_mod	Mod	-
Moments	tf.nn.moments	-	-	-
Mul	tf.math.multiply	tfl.mul	Mul	Eltwise
Negative	tf.math.negative	tfl.neg	Neg	-
NMS	tf.image.non_max_suppression	-	NonMaxSuppression	-
	tf.raw_ops.NonMaxSuppressionV4	tfl.non_max_suppression_v4		-
	tf.image.non_max_suppression_with_scores			-
OneHot	tf.one_hot	tfl.one_hot	OneHot	-
Pad	tf.pad	tfl.pad	Pad	-
		tfl.padv2		-
		tfl.mirror_pad		-
Pow	tf.math.pow	tfl.pow	Pow	Power
Prelu	tf.keras.layers.PReLU	tfl.prelu	Prelu	PReLU
Reciprocal	tf.math.reciprocal	-	Reciprocal	-
ReduceAll	tf.math.reduce_all	tfl.reduce_all	-	-
ReduceAny	tf.math.reduce_any	tfl.reduce_any	-	-
ReduceL1	-	-	ReduceL1	-
ReduceL2	-	-	ReduceL2	-
ReduceMax	tf.math.reduce_max	tfl.reduce_max	ReduceMax	-
ReduceMean	tf.math.reduce_mean	tfl.mean	ReduceMean	Reduction
ReduceMin	tf.math.reduce_min	tfl.reduce_min	ReduceMin	-

IR	TensorFlow	TensorFlow-Lite	ONNX	Caffe
ReduceProd	tf.math.reduce_prod	tfl.reduce_prod	ReduceProd	–
ReduceSum	tf.math.reduce_sum	tfl.sum	ReduceSum	Reduction
ReduceUnbiasVariance	–	–	–	–
ReduceVariance	tf.math.reduce_variance	–	–	–
Relu	tf.nn.relu	tfl.relu	Relu	ReLU
Relu6	tf.nn.relu6	tfl.relu6	–	–
Repeat	–	–	–	–
Reshape	tf.reshape	tfl.reshape	Reshape	Reshape
	tf.expand_dims	tfl.expand_dims	Expand	–
	tf.keras.layers.Flatten	–	Flatten	Faltten
	tf.squeeze	tfl.squeeze	Squeeze	–
Resize	tf.compat.v1.image.resize_bilinear	tfl.resize_bilinear	Resize	–
	tf.compat.v1.image.resize_nearest_neighbor	tfl.resize_nearest_neighbor		–
ReverseSequence	tf.reverse_sequence	tfl.reverse_sequence	ReverseSequence	–
RgbToYuv	–	–	–	–
RightShift	tf.bitwise.right_shift	–	BitShift	–
RoiAlign	–	–	RoiAlign	–
Round	tf.math.round	tfl.round	Round	–
Rsqrt	tf.math.rsqrt	tfl.rsqrt	–	–
ScatterElements	–	–	ScatterElements	–
ScatterND	tf.scatter_nd	tfl.scatter_nd	ScatterND	–
	tf.tensor_scatter_nd_add			–
	tf.tensor_scatter_nd_update			–
SegmentSum	tf.math.segment_sum	tfl.segment_sum	–	–
Selu	tf.nn.selu	–	Selu	–
Shrink	–	–	Shrink	–
Sigmoid	tf.math.sigmoid	tfl.logistic	Sigmoid	Sigmoid

IR	TensorFlow	TensorFlow-Lite	ONNX	Caffe
Sign	tf.math.sign	–	Sign	–
Silu	–	–	Sigmoid + Mul	–
Sine	tf.math.sin	tfl.sin	Sin	–
Sinh	tf.math.sinh	–	Sinh	–
Slice	tf.slice	tfl.slice	Slice	Slice
	tf.strided_slice	tfl.strided_slice		
Softmax	tf.nn.softmax	tfl.softmax	Softmax	Softmax
Softplus	tf.math.softplus	–	Softplus	–
Softsign	tf.nn.softsign	–	Softsign	–
SpaceToBatch	tf.space_to_batch	tfl.space_to_batch_nd	–	–
SpaceToDepth	tf.nn.space_to_depth	tfl.space_to_depth	SpaceToDepth	–
Split	tf.split	tfl.split	Split	–
Sqrt	tf.math.sqrt	tfl.sqrt	Sqrt	–
Square	tf.math.square	tfl.square	–	–
SquaredDifference	tf.math.squared_difference	tfl.squared_difference	–	–
Sub	tf.math.subtract	tfl.sub	Sub	–
Tan	tf.math.tan	–	Tan	–
Tanh	tf.math.tanh	tfl.tanh	Tanh	TanH
ThresholdedRelu	tf.keras.layers.ThresholdedReLU	–	ThresholdedRelu	–
Tile	tf.tile	tfl.tile	Tile	–
TopK	tf.math.top_k	tfl.topk_v2	TopK	ArgMax
Transpose	tf.transpose	tfl.transpose	Transpose	–
UpsampleByIndex	–	–	–	–
Where	tf.where	tfl.select	–	–
YuvToRgb	–	–	–	–
ZeroFraction	tf.math.zero_fraction	–	–	–

3 AIPU Operators

This section describes all officially supported built-in operators on AIPU.

Except for special cases, all input and output tensors must meet the following conditions:

- shape_size < = 64MB
- Dim[0] < = 32.

3.1 Abs

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-

3.2 AccidentalHits

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 1920] Dim[1] ∈ [1, 16384] Input1: Supports int16 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 16384]	-

3.3 Acos

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

3.4 Acosh

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

3.5 Add

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input.	Input format can be: [[N,C],[C]], [[N,C],[1]], [[N,C],[N,1]], [[N,H,C],[C]], [[N,H,C],[1]], [[N,H,C],[H,1]], [[N,H,C],[N,1,1]], [[N,H,W,C],[C]], [[N,H,W,C],[1]], [[N,H,W,C],[H,1,1]], [[N,H,W,C],[N,1,1,1]], and the order of the two inputs can be swapped.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.6 ArgMax

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
axis	[0, 3]	-
select_last_index	{true, false}	-

3.7 ArgMin

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
axis	[0, 3]	-
select_last_index	{true, false}	-

3.8 Asin

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

3.9 Asinh

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

3.10 AveragePooling2D

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
kernel_x / kernel_y	[1, 65]	-
stride_x / stride_y	[1, 6]	The following must be true: • stride_x < = kernel_x • stride_y < = kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
pad_top / pad_bottom pad_left / pad_right	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
dilation_x / dilation_y	{1}	-
ceil_mode	{true, false}	-
count_include_pad	{true, false}	-
method	{AVG}	-

3.11 AveragePooling3D

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8 and uint8 data types. Supports 5-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 2048]	-
kernel_x/y/z	[1, 17]	kernel_x * kernel_y * kernel_z should be less than or equal to 256.
stride_x/y/z	[1, 8]	The following must be true: • stride_x < = kernel_x • stride_y < = kernel_y • stride_z < = kernel_z Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2

Parameter	Valid value or range	Comment
pad_x_begin/end pad_y_begin/end pad_z_begin/end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
dilation_x/y/z	{1}	-
ceil_mode	{true, false}	-
count_include_pad	{true, false}	-
method	{AVG}	-

3.12 BNLL

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

3.13 BasicLSTM

Parameter	Valid value or range	Comment
Number of input tensors	3	-
Number of output tensors	1-3	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 3071] Dim[2] ∈ [1, 3072] Input1_2: Supports int8 and int16 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 3072]	• Input0: {X} • Input1: {HO} • Input2: {CO}

Parameter	Valid value or range	Comment
output shape(s)	Output0_2: Supports int8 and int16 data types.	Output: {Y}, {H}, {C}, {Y, H}, {Y, C}, {H, C} or {Y, H, C}, which depends on out_sequence.
out_sequence	{{Y}, {H}, {C}, {Y, H}, {Y, C}, {H, C}, {Y, H, C}}	-
activations	{Tanh, Sigmoid}	-
direction	{Forward, Reverse}	-

3.14 BatchNormalization

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	• For 2-dims, axis == 1 • For 3-dims, axis == 2 • For 4-dims, axis == 3
output shape(s)	Supports int8 and int16 data types.	-
axis	[1, input_dims-1]	-

3.15 BatchToSpace

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
block_size_x/y	[1, 16]	-

Parameter	Valid value or range	Comment
crop_left / crop_right / crop_top / crop_bottom	[0, 16]	-

3.16 BiasAdd

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-

3.17 CTCGreedyDecoder

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] Dim[2] ∈ [1, 8192] Input1: Supports uint16 data type Supports 1-dimensional input. Dim[0] ∈ [1, 32]	-
merge_repeated	{true}	-

3.18 Cast

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
to_dtype	{int8, uint8, int16, uint16}	-

3.19 Ceil

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

3.20 Celu

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

Parameter	Valid value or range	Comment
method	{CELU}	-

3.21 ChannelShuffle

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
group	[1, 4096]	-
splits	[1, 16]	1 or input_shape[-1]/group

3.22 Clip

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
method	{CELU}	-

3.23 Compress

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports int8 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 4096]	The shape of input1 should be less than or equal to input0.shape[axis].
axis	[0, 3]	-

3.24 Concat

Parameter	Valid value or range	Comment
Number of input tensors	2-20	-
input shape(s)	Input0_19: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
axis	-	axis ∈ [-1, input_dims-1]

3.25 Constant

Parameter	Valid value or range	Comment
Number of input tensors	0	-
input shape(s)	-	-
weights_shape	-	Any length and dimension, as long as the total size meets the requirements.

3.26 ConvTranspose2D

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor.
kernel_x / kernel_y	[1, 64]	-
stride_x / stride_y	[1, 16]	-
pad_top / pad_bottom / pad_left / pad_right	[0, 16]	-
dilation_x / dilation_y	[1, 16]	If dilation_x/y != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y < = 1G
group	{1}	-
with_activation	{NONE, RELU, CLIP, PRELU, LEAKYRELU}	-

3.27 ConvTranspose3D

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 5-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	The input must be an NDHWC format tensor. Dims[4] * Dims[2]: [1, 4096]
kernel_x / kernel_y	[1, 11]	-
kernel_z	[1, 11]	The following must be true: (Input_c * kernel_z % 32 == 0 and 32< =input_c * kernel_z < =4096) Or: Input_c * kernel_z ∈ {1, 3, 4}

Parameter	Valid value or range	Comment
stride_x / stride_y / stride_z	[1, 6]	The following must be true: • stride_x < = kernel_x • stride_y < = kernel_y • stride_z < = kernel_z Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2
pad_x_begin/end pad_y_begin/end pad_z_begin/end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
dilation_x / dilation_y / dilation_z	{1}	-
group	{1}	-
with_activation	{NONE, RELU, CLIP, LEAKYRELU}	-

3.28 Convolution2D

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor.
output shape(s)	Supports int8 and int16 data types.	The output must be an NHWC format tensor.
kernel_x / kernel_y	[1, 64]	-
stride_x / stride_y	[1, 16]	-
pad_top / pad_bottom / pad_left / pad_right	[0, 16]	-
dilation_x / dilation_y	[1, 16]	If dilation_x/y != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y < = 1G
group	{1}	-
with_activation	{NONE, RELU, CLIP, PRELU, LEAKYRELU}	-

3.29 Convolution3D

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 5-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	The input must be an NDHWC format tensor. Dims[4] * Dims[2]: [1, 4096]
kernel_x / kernel_y	[1, 11]	-
kernel_z	[1, 11]	The following must be true: (Input_c * kernel_z % 32 == 0 and 32< =input_c * kernel_z < =4096) Or: Input_c * kernel_z ∈ {1, 3, 4}
stride_x / stride_y / stride_z	[1, 6]	The following must be true: • stride_x < = kernel_x • stride_y < = kernel_y • stride_z < = kernel_z Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2
pad_x_begin/end pad_y_begin/end pad_z_begin/end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
dilation_x / dilation_y / dilation_z	{1}	-
group	{1}	-
with_activation	{NONE, RELU, CLIP, LEAKYRELU}	-

3.30 Cosh

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

3.31 Cosine

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

3.32 Count

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports uint8 and uint16 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384]	-
min	[0, 65534]	Any value in the range of input data type. (max > min)
max	[1, 65535]	Any value in the range of input data type. (max > min)
nbins	[1, 4096]	-
discrete	{true}	-

3.33 Crelu

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
method	{CRELU}	-
axis	-	axis = input_dims-1 or -1

3.34 Crop

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	-

Parameter	Valid value or range	Comment
crops	crops[0][0] ∈ [0, 30] crops[0][1] ∈ [1, 31] crops[1][0] ∈ [0, 98] crops[1][1] ∈ [1, 99] crops[2][0] ∈ [0, 1078] crops[2][1] ∈ [1, 1079] crops[3][0] ∈ [0, 1918] crops[3][1] ∈ [1, 1919] crops[4][0] ∈ [0, 4094] crops[4][1] ∈ [1, 4095]	crops[i][0] should be less than crops[i][1].

3.35 CropAndResize

Parameter	Valid value or range	Comment
Number of input tensors	3	-
input shape(s)	Input0: Supports uint8 data type Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports uint16 data type Supports 2-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [4, 4] Input2: Supports uint8 data type Supports 1-dimensional input. Dim[0] ∈ [1, 16384]	Number of ROIs
crop_size	crop_size[0] ∈ [0, 1080] crop_size[1] ∈ [1, 1920]	-
method	{BILINEAR, NEAREST}	-
extrapolation_value	{0, 1}	-

3.36 DataStride

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
kernel_x / kernel_y	[1, 16]	kernel_x/y must be smaller than or equal to stride_x/y.
stride_x / stride_y	[1, 16]	stride_x/y should be smaller than or equal to input_w/h. Only supports kernel_x == kernel_y, stride_x == stride_y.

3.37 DepthToSpace

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
block_size_x / block_size_y	[1, 16]	Only supports block_size_x == block_size_y.

3.38 DepthwiseConvolution

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor.
kernel_x / kernel_y	[1, 64]	-
stride_x / stride_y	[1, 16]	-
pad_top / pad_bottom / pad_left / pad_right	[0, 16]	-
dilation_x / dilation_y	[1, 16]	-

Parameter	Valid value or range	Comment
group	[1, 16384]	The following must be true: Input_dim[3] == group
multiplier	[1, 4096]	The following must be true: Output_dim[3] == Input_dim[3] * multiplier
with_activation	{NONE, RELU, CLIP, PRELU, LEAKYRELU}	-

3.39 Div

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0_1: Supports uint8 data type. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-

3.40 ElementwiseAdd

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0_1: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{ADD}	-
with_activation	{NONE, RELU, LEAKYRELU}	-

3.41 ElementwiseMax

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0_1: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
method	{MAX}	-
with_activation	{NONE, RELU, LEAKYRELU}	-

3.42 ElementwiseMin

Parameter	Valid value or range	Comment
Number of input tensors	2	-

Parameter	Valid value or range	Comment
input shape(s)	Input0_1: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
method	{MIN}	-
with_activation	{NONE, RELU, LEAKYRELU}	-

3.43 ElementwiseMul

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0_1: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
method	{MUL}	-
with_activation	{NONE, RELU, LEAKYRELU}	-

3.44 ElementwiseSub

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0_1: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
method	{SUB}	-
with_activation	{NONE, RELU, LEAKYRELU}	-

3.45 Elu

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{ELU}	-

3.46 Exp

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.47 Filter

Parameter	Valid value or range	Comment
Number of input tensors	2-10	-
Number of output tensors	2	-
input shape(s)	Input0_8: Supports int8 and int16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input9: Supports int8 and int16 data types. Supports 1-dimensional input.	Input9: A vector with the same length as the specified axis in input 0
output shape(s)	Output0_1: Supports int8 and int16 data types.	Output0: The same shape as input 0
axis	{0}	-
num	[1, 8]	-

3.48 Floor

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
Output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.49 FullyConnected

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384]	The input must be an NC format tensor.
output shape(s)	Supports int8, int16, uint8, and uint16 data types.	The output must be an NC format tensor.

3.50 GRUv1

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	2	-

Parameter	Valid value or range	Comment
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 3-dimensional input. Dim[0], Dim[2] ∈ [1, 16384] Dim[1] ∈ [1, 4096] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [1, 8192]	-
output shape(s)	Output0_1: Supports int8, uint8, int16, and uint16 data types.	The output dtype is equal to input1 dtype.
out_sequence	{{H}, {Hn}, {H, Hn}}	-
activations	{Relu, Tanh, Sigmoid, Affine, LeakyRelu, ThresholdedRelu, HardSighmoid, Elu, Softsign, Softplus}	-
direction	{forward}	-

3.51 GRUv3

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 3-dimensional input. Dim[0], Dim[2] ∈ [1, 16384] Dim[1] ∈ [1, 4096] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [1, 8192]	-
output shape(s)	Output0_1: Supports int8, uint8, int16, and uint16 data types.	The output dtype is equal to input1 dtype.
out_sequence	{{H}, {Hn}, {H, Hn}}	-
activations	{Relu, Tanh, Sigmoid, Affine, LeakyRelu, ThresholdedRelu, HardSighmoid, Elu, Softsign, Softplus}	-

Parameter	Valid value or range	Comment
direction	{forward}	-

3.52 Gather

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	-	axis ∈ [0, input0_dims]
Batch_dims	-	Batch_dims ∈ [0, axis]

3.53 GatherElements

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0_1: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	-	Batch_dims ∈ [0, input0_dims]

3.54 GatherND

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] Dim[2] ∈ [1, 4096] Dim[3] ∈ [1, 4096] Input1: Supports uint16 data type. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	• Rank(input1) - batch_dim = 2 • Batch_dims < min(rank(input0), rank(input1)) • Input1.shape[-1] < = rank(input0) - batch_dims • Input0.shape[:batch_dims] = input1.shape[:batch_dims]
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	[0, 3]	-

3.55 Gemm

Parameter	Valid value or range	Comment
Number of input tensors	3	-
Number of output tensors	1	-
input shape(s)	Input0_2: Supports int8 and uint8 data types. Supports 2-dimensional input. Dim[0], Dim[1] ∈ [1, 1920]	The matrix size must satisfy the matrix multiplication rule.
output shape(s)	Supports int8 and uint8 data types.	The matrix size must satisfy the matrix multiplication rule.

Parameter	Valid value or range	Comment
trans_a / trans_b	{True, False}	-
alpha / beta	[1, 19]	-

3.56 GridSample

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 4096] Input1: Supports int16 data type. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 4096] Dim[3] ∈ [2, 2]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{NEAREST, BILINEAR}	-
padding_mode	{ZEROS, BORDER}	-
align_corners	{True, False}	-

3.57 GroupConvolution

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and int16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor.
output shape(s)	Supports int8 and int16 data types.	The output must be an NHWC format tensor.
kernel_x / kernel_y	[1, 64]	-
stride_x / stride_y	[1, 16]	-
pad_top / pad_bottom / pad_left / pad_right	[0, 16]	-

Parameter	Valid value or range	Comment
dilation_x / dilation_y	[1, 16]	If dilation_x/y != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y < = 1G
group	{1, 32}	The following must be true: • Output_dim[3] % group == 0 • Input_dim[3] % group == 0 • group > 1

3.58 GroupNorm

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8 and uint8 data type. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
axis	{3}	-
group	[1, 4096]	The following must be true: • input.shape[axis] % group == 0 • input.shape[axis] % 4 == 0

3.59 HardSigmoid

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
method	{HARDSIGMOID}	-

3.60 HardSwish

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.61 InTopK

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384] Input1: Supports uint8 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384]	-
output shape(s)	Supports int8 and uint8 data types.	-
k	[1, 16384]	-
axis	{-1}	-

3.62 InstanceNormalization

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	The input must be an NHWC format tensor.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	The output must be an NHWC format tensor.

3.63 L1Pooling2D

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
kernel_x / kernel_y	[1, 65]	-
stride_x / stride_y	[1, 6]	The following must be true: • stride_x < = kernel_x • stride_y < = kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
pad_top / pad_bottom / pad_left / pad_right	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
method	{L1}	-

3.64 L2Pooling2D

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
kernel_x / kernel_y	[1, 65]	-
stride_x / stride_y	[1, 6]	The following must be true: • stride_x < = kernel_x • stride_y < = kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
pad_top / pad_bottom / pad_left / pad_right	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
method	{L2}	-

3.65 LRN

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{ACROSS_CHANNELS, WITHIN_CHANNEL}	-
size	[1, 64]	-
bias	[1, 64]	-
alpha	[1, 64]	-
beta	[0.0, 1.0]	Left open right open interval

3.66 LayerNormalization

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	The input must be an NHWC format tensor.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	The output must be an NHWC format tensor.
method	{WITH_GAMMA_BETA, WITHOUT_GAMMA_BETA}	-
axis	{1, 2, 3}	-

3.67 LeakyRelu

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{LEAKYRELU}	-

3.68 LeftShift

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports uint8 data type. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	Broadcast is not supported, and both inputs should share the same shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
direction	{LEFT}	-

3.69 Log

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.70 LogSoftmax

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 3968] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 3968] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 3968]	-
output shape(s)	Supports int8 data type.	-
axis	[-1, input_dims - 1]	-

3.71 Logical

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0_1: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	The shape of the second input is equal to the shape of the first input. Input1 may not exist.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{EQUAL, NOT_EQUAL, GREATER, GREATER_EQUAL, LESS, LESS_EQUAL, XOR, NOT}	If method=='XOR', len(bottoms) ==1 & len(scale_value) == 1.

3.72 MatMul

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0_1: Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	The range of Dim[0] * Dim[1] ∈ [1,16384], and must be equal to Dim[0] * Dim[1] in the input shape. The matrix size must satisfy the matrix multiplication rule.

3.73 MaxPooling2D

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
Output shape(s)	Supports int8 and int16 data types.	-
kernel_x / kernel_y	[1, 65]	-
stride_x / stride_y	[1, 6]	The following must be true: • stride_x < = kernel_x • stride_y < = kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
pad_top / pad_bottom pad_left / pad_right	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
method	{MAX}	-

3.74 MaxPooling3D

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types. Supports 5-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	-
kernel_x/y/z	[1, 1080]	-
kernel_y	[1, 1920]	-
kernel_z	[1, 100]	-
stride_x	[1, 1080]	-
stride_y	[1, 1920]	-
pad_x_begin/end pad_y_begin/end pad_z_begin/end	[0, 16]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
ceil_mode	{False}	-

Parameter	Valid value or range	Comment
count_include_pad	{False}	-
method	{MAX}	-

3.75 MaxPoolingWithArgMax

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	2	-
input shape(s)	Supports int8 data type. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
Output shape(s)	Output0: Supports int8 data type. Output1: Supports int32 data type.	-
kernel_x / kernel_y	[1, 17]	-
stride_x / stride_y	[1, 17]	The following must be true: • stride_x < = kernel_x • stride_y < = kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
pad_top / pad_bottom pad_left / pad_right	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
dilation_x / dilation_y	{1}	-
storage_order	{0, 1}	-

3.76 MaxRoiPool

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports uint16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [5, 5]	-
output shape(s)	Supports int8 data type.	-
pooled_shape	pooled_shape[0] ∈ [1, 1080] pooled_shape[1] ∈ [1, 1920]	-
spatial	spatial[0], spatial[1] ∈ [1, 65535]	-

3.77 MaxUnpool

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0_1: Supports int8 and uint8 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 16] Dim[1], Dim[2], Dim[3] ∈ [1, 4096]	-
flattem_dim	{HW, HWC, NHWC, NCHW}	-
storage_order	{0, 1}	-
output_shape	output_shape[0] ∈ [1, 16] output_shape[1], output_shape[2], output_shape[3] ∈ [1, 4096]	-

3.78 MeanVarianceNormalization

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-

Parameter	Valid value or range	Comment
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	{{1, 2, 3}, {1, 2}}	-
epsilon	[0.0, 64.0]	Optional, left open right closed interval

3.79 Mish

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{MISH}	-

3.80 Mod

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8 and uint8 data types.	-

Parameter	Valid value or range	Comment
fmod	{0, 1}	Optional

3.81 Moments

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	{{0}, {1}, {2}, {3}, {0, 1}, {0, 2}, {0, 3}, {1, 1}, {1, 2}, {1, 3}, {0, 1, 2}, {0, 1, 3}, {0, 2, 3}, {1, 2, 3}, {0, 1, 2, 3}}	-
keepdims	{True, False}	Optional

3.82 Mul

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input.	Input format can be: [[N,C],[C]], [[N,C],[1]], [[N,C],[N,1]], [[N,H,C],[C]], [[N,H,C],[1]], [[N,H,C],[H,1]], [[N,H,C],[N,1,1]], [[N,H,W,C],[C]], [[N,H,W,C],[1]], [[N,H,W,C],[H,1,1]], [[N,H,W,C],[N,1,1,1]], and the order of the two inputs can be swapped.

Parameter	Valid value or range	Comment
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.83 NMS

Parameter	Valid value or range	Comment
Number of input tensors	4	-
Number of output tensors	4	-
input shape(s)	Input0: Supports int16 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [4, 4] Input1: Supports uint16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Input2: Supports uint16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1] Input3: Supports uint8 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [4, 4]	-
output shape(s)	Output0_1_3: Supports int16 data type. Output2: Supports uint8 data type.	-
iou_threshold	{0, 16384}	-
center_point_box	{0, 1}	Optional

3.84 Negative

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8 and int16 data types.	-

3.85 OneHot

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	{-1, 3}	-
depth	[1, 16384]	-
on_value / off_value	[0, 65535]	-

3.86 Pad

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	-
crops	crops[0][0], crops[0][1], crops[1][0], crops[1][1], crops[2][0], crops[2][1], crops[3][0], crops[3][1], crops[4][0], crops[4][1] ∈ [0, 16]	-
method	{CONSTANT, REFLECT, SYMMETRIC}	-

3.87 Pow

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0_1: Supports int8 and uint8 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
exponent	[1, 9]	-

3.88 Prelu

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
method	{PRELU}	-
negative_slope_type	{uint16}	-
negative_slope_shape	-	== input0_shape[-1]

3.89 Reciprocal

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
method	{NO_CONSTANT}	-

3.90 ReduceAll

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	-
axis	-	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{ALL}	-

3.91 ReduceAny

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	-

Parameter	Valid value or range	Comment
axis	–	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{ANY}	–

3.92 ReduceL1

Parameter	Valid value or range	Comment
Number of input tensors	1	–
Number of output tensors	1	–
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	–
axis	–	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{L1}	–

3.93 ReduceL2

Parameter	Valid value or range	Comment
Number of input tensors	1	–
Number of output tensors	1	–

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	-
axis	-	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{L2}	-

3.94 ReduceMax

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	-

Parameter	Valid value or range	Comment
axis	–	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{MAX}	–

3.95 ReduceMean

Parameter	Valid value or range	Comment
Number of input tensors	1	–
Number of output tensors	1	–
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	–
axis	–	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{MEAN}	–

3.96 ReduceMin

Parameter	Valid value or range	Comment
Number of input tensors	1	–
Number of output tensors	1	–

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	-
axis	-	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{MIN}	-

3.97 ReduceProd

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	-

Parameter	Valid value or range	Comment
axis	-	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{PROD}	-

3.98 ReduceSum

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	-
axis	-	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{SUM}	-

3.99 ReduceUnbiasedVariance

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	-
axis	-	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{UNBIASED_VARIANCE}	-

3.100 ReduceVariance

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports uint8 and uint16 data types.	-

Parameter	Valid value or range	Comment
axis	–	The axis can be a number or a set of numbers. All the values are ∈ [-1, input_dims - 1].
method	{VARIANCE}	–

3.101 Relu

Parameter	Valid value or range	Comment
Number of input tensors	1	–
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	–
method	{RELU}	–

3.102 Relu6

Parameter	Valid value or range	Comment
Number of input tensors	1	–
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	–
method	{CRELU6}	–

3.103 Repeat

Parameter	Valid value or range	Comment
Number of input tensors	2	–

Parameter	Valid value or range	Comment
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 16] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 16] Dim[1], Dim[2], Dim[3] ∈ [1, 16384] Input1: Supports uint16 data type. Supports 1-dimensional input.	[input1_shape = input0_dims[axis]] if axis else [data_size of input0]
axis	[-1, input0_dims - 1] or None	-

3.104 Reshape

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	

3.105 Resize

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
ratio_x / ratio_y	[1, 8]	-
method	{NEAREST, BILINEAR}	-
mode	{ALIGN_CORNERS, HALF_PIXEL, ASYMMETRIC, PYTORCH_HALF_PIXEL, TF_HALF_PIXEL_FOR_NN}	-
nearest_mode	{FLOOR, CEIL, ROUND_PREFER_CEIL, SIMPLE}	-
interp_shift_value	[0, 13]	This parameter is required for TPC, but not for AIFF.

3.106 ReverseSequence

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] Input1: Supports uint16 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 1]	The input1 cannot be provided.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
batch_axis	{0, 1}	-
time_axis	{0, 1, 2}	The value 2 only can be used for 3-dims input0, and time_axis cannot be equal to batch_axis.

3.107 RgbToYuv

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
format	{I420}	-
bits	{8}	-
conversion	{BT709}	-
coefficient	{[0, 0, 0, 0, 128, 128, 218, 732, 74, -118, -395, 512, 512, -465, -47]}	-
coefficient_dtype	{int16}	-
coefficient_shift	{10}	-

3.108 RightShift

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports uint8 data type. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	Broadcast is not supported, and both inputs should share the same shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	–
direction	{RIGHT}	–

3.109 RoiAlign

Parameter	Valid value or range	Comment
Number of input tensors	2	–

Parameter	Valid value or range	Comment
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1], Dim[2] Dim[3] ∈ [1, 16384] Input1: Supports uint16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [5, 5]	-
method	{AVG, MAX}	-
sample	{[0, 8], [0, 8]}	-
coordinate_transformation_mode	{OUTPUT_HALF_PIXEL, HARF_PIXEL}	

3.110 Round

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

3.111 Rsqrt

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-

3.112 ScatterElements

Parameter	Valid value or range	Comment
Number of input tensors	3	-
Number of output tensors	1	-

input shape(s)	Input0:
	Supports int8, uint8, int16, and uint16 data types.
	Supports 2, 3, 4, 5-dimensional input.
	For 2-dims:
	Dim[0] ∈ [1, 32]
	Dim[1] ∈ [1, 16384]
	For 3-dims:
	Dim[0] ∈ [1, 32]
	Dim[1], Dim[2] ∈ [1, 16384]
	For 4-dims:
	Dim[0] ∈ [1, 32]
	Dim[1], Dim[2], Dim[3] ∈ [1, 16384]
	For 5-dims:
	Dim[0] ∈ [1, 32]
	Dim[1], Dim[2], Dim[3], Dim[4] ∈ [1, 16384]
	Input1_2:
	Supports int8, uint8, int16, and uint16 data types.
	Supports 2-dimensional input.
		input1_dim = input0_dim
		input1_shape < = input0_shape
		input2_shape = input1_shape

output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.113 ScatterND

Parameter	Valid value or range	Comment
Number of input tensors	3	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3], Dim[4] ∈ [1, 16384] Input1_2: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input.	input1_shape[-1] < dim(input0) input2_shape= input1_shape[:-1] + input0_shape[input0_dim-input1_shape[-1]:]
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.114 SegmentSum

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384] Input1: Supports uint16 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 32]	The input1 shape must be equal to input0_shape[0].

Parameter	Valid value or range	Comment
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	–
method	{SUM}	–

3.115 Selu

Parameter	Valid value or range	Comment
Number of input tensors	1	–
Number of output tensors	1	–
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	–
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	–
method	{SELU}	

3.116 Shrink

Parameter	Valid value or range	Comment
Number of input tensors	1	–
Number of output tensors	1	–
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	–
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	–
method	{SHRINK}	–

3.117 Sigmoid

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SIGMOID}	

3.118 Sign

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.119 Silu

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SILU}	-

3.120 Sine

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.121 Sinh

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.122 Slice

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
begin / end	-	The value of begin/end must be less than the total shape of input and greater than 0.

3.123 Softmax

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 3968] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 3968] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 3968]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	{-1}	-

3.124 Softplus

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SOFTPLUS}	

3.125 Softsign

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SOFTSIGN}	-

3.126 SpaceToBatch

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
block_size_x/y	[1, 16]	-
pad_top / pad_bottom / pad_left / pad_right	[0, 16]	-

3.127 SpaceToDepth

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
block_size_x/y	[1, 16]	-

Split

Parameter	Valid value or range	Comment
Number of input tensors	1	–
Number of output tensors	2–16	–
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	–
output shape(s)	Output0_15: Supports int8, uint8, int16, and uint16 data types.	–
splits	[1, 16]	–
axis	[0, 3]	–

3.128 Sqrt

Parameter	Valid value or range	Comment
Number of input tensors	1	–
Number of output tensors	1	–
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	–
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	–

3.129 Square

Parameter	Valid value or range	Comment
Number of input tensors	1	–

Parameter	Valid value or range	Comment
Number of output tensors	1	–
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	–
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	–

3.130 SquaredDifference

Parameter	Valid value or range	Comment
Number of input tensors	2	–
Number of output tensors	1	–
input shape(s)	Input0_1: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	–
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	–

3.131 Sub

Parameter	Valid value or range	Comment
Number of input tensors	2	–

Parameter	Valid value or range	Comment
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input.	Input format can be: [[N,C],[C]], [[N,C],[1]], [[N,C],[N,1]], [[N,H,C],[C]], [[N,H,C],[1]], [[N,H,C],[H,1]], [[N,H,C],[N,1,1]], [[N,H,W,C],[C]], [[N,H,W,C],[1]], [[N,H,W,C],[H,1,1]], [[N,H,W,C],[N,1,1,1]], and the order of the two inputs can be swapped.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.132 Tan

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.133 Tanh

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{TANH}	-

3.134 ThresholdedRelu

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{THRESHOLDEDRELU}	-

3.135 Tile

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
repeats	-	The shape of repeats is less than or equal to the output shape.

3.136 TopK

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-
Number of output tensors	2	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 1, 2, 3, 4-dimensional input. Input1: Supports int8 and uint8 data types. Supports 1-dimensional input.	Input1 is optional.
output shape(s)	Output0_1: Supports int8 and uint8 data types.	For d in [0, shape_dims-1] and d != axis, Out_shape[d] == input0_shape[d]
k	[1, 16384]	-
axis	{-1}	-
largest	{true}	-
sorted	{true, false}	-
select_index	{last}	-

3.137 Transpose

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
perm	[0, 3]	-

3.138 UpsampleByIndex

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0_1: Supports int8 and uint8 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 16] Dim[1], Dim[2], Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8 and uint8 data types.	-
flattem_dim	{HW, HWC, NHWC, NCHW}	-
storage_order	{0, 1}	-
output_shape	output_shape[0] ∈ [1, 16] output_shape[1], output_shape[2], output_shape[3] ∈ [1, 4096]	-

3.139 Where

Parameter	Valid value or range	Comment
Number of input tensors	1-3	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0_2: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1], Dim[2], Dim[3] ∈ [1, 16384]	The number of inputs cannot be 2.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

3.140 YuvToRgb

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports uint8 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 6220800]	-
output shape(s)	Supports uint8 data type.	-
format	{I420}	-
bits	{8}	-
conversion	{BT601, BT709, BT2020, SELF}	-
coefficient	-	1-d array. The length is 15.
coefficient_dtype	{int16}	-

3.141 ZeroFraction

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 1, 2, 3, 4-dimensional input. For 1-dims: Dim[0] ∈ [1, 4096] For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

4 SlimAI Operators

The following table lists all ONNX operators supported by SlimAI.

Operator	Inputs	Outputs	Restrictions
Abs	float		Not supported in QuantByPass.
Add	int32, int64, float		
ArgMax	float	int32, int64	axis: [1..3] Can only have a single output consumer. Must be the last node in the network. Input must be a 4D blob including batch.
AveragePool	float		kernel_shape[i]: [2..128] count_include_pad: Not supported stride[i]: [1..2] stride: All same values auto_pad: VALID, SAME_UPPER, SAME_LOWER
BatchNormalization	float		
InstanceNormalization	float		
Ceil	float		Not supported in QuantByPass.
Clip	float		
Concat	float		
Constant			
ConstantOfShape	int32, int64, float		
Conv	float		Supported strides: 1, 2, 4 Supported kernel sizes: M x N (any combination of kernel sizes up to 16x16) dilation: same for both axis, stride must be 1 when dilation is > 1 auto_pad: VALID, SAME_UPPER, SAME_LOWER
Cast			Some constraints, Float to bool
ConvTranspose	float		dilation: 1 group: 1 or group = output channels (Depthwise Deconvolutions) auto_pad: VALID, SAME_UPPER, SAME_LOWER Deconvolution is broken into smaller convolutions which multiply non-zero activations only. The smaller kernels must follow the restrictions of convolution.
Cos	float		Not supported in QuantByPass.

Operator	Inputs	Outputs	Restrictions
Clip (V10, V11)	float (V10 args, V11 inputs) **min**: constant **max**: constant		**Untested:** min and max default values.
DepthToSpace (V10, V11)			
Div	float		Constant dividend and tensor diviser not supported.
Dropout			
Elu	float		Not supported in QuantByPass.
Exp	float		
Expand			
Flatten			axis: only support 1
Floor	float		Not supported in QuantByPass
Gather	int32, int64, float		axis: 0 or 1
Gemm	float		
GlobalAveragePool	float		
Greater	float		
GRU	float	Float (concatenated hidden output Y only)	clip, activation_alpha, activation_beta attributes are unsupported. sequence_lens must be fixed/constant value (or is inferred to be constant value based on the shape of input). Activations supported are only Relu, TanH, and Sigmoid. linear_before_reset is supported with unrolled implementation but has not been tested. Really long sequence_lens may hit memory limitations which depend on the overall size of the network and number of GRU nodes. Not supported in QuantByPass.
			DSP only Monolithic GRU kernel has same constraints as lowered kernel and some additional limitations. It supports only TanH, and Sigmoid nodes for activations, Fully Connected Layers wherever applicable and also both forward and reverse directions. It has been tested with only Sigmoid for update and reset activation gates and Tanh for hidden-state activation gates.
Identity			

Operator	Inputs	Outputs	Restrictions
LRN	float		Kernel size: 3, 5, 7, 9, 11 (along depth only)
LeakyRelu	slope: [0, 1]		
Less	int32, int64, float	bool	
Log	float		Not supported in QuantByPass.
LSTM	float	float	clip, activation_alpha, activation_beta attributes are unsupported. sequence_lens must be fixed/constant value (or is inferred to be constant value based on the shape of input). Activations supported are only Relu, TanH, and Sigmoid. input_forget and peephole features are supported but untested. Really long sequence_lens may hit memory limitations which depend on the overall size of the network and number of LSTM nodes. Not supported in QuantByPass.
			DSP only Monolithic LSTM kernel has same constraints as lowered kernel and some additional limitations. It supports only TanH, and Sigmoid nodes for activations, Fully Connected Layers wherever applicable and also both forward and reverse directions. It been tested with Sigmoid for input, output and forget activation gates and Tanh for candidate and hidden-state activation gates. Monolithic kernel does not support input_forget and peephole attributes.
MatMul	float		
Max	float		Max with constant scalar
MaxPool	float		kernel_shape: 2D Supported strides: 1, 2 Supported kernel sizes: NxN (kernel sizes up to 16x16) Global Max Pooling up to 16x16
Min	float		Min with constant scalar
Mul	int32, int64, float		No Broadcast support
Neg	float		
NonZero			Constant inputs only
PRelu	float		Only when slope is a splat or is a broadcasted channel dimension.
Pad			mode: CONSTANT, support only positive pads and padding value of zero. Mode: REFLECT, support positive pads only.

Operator	Inputs	Outputs	Restrictions
Pow	float		One of the inputs (either base or power) must be a uniform constant. Not supported in QuantByPass.
Reciprocal	float		Not supported in QuantByPass.
ReduceMean	float		
ReduceSum	float		
ReduceSumSquare	float		
ReduceMin	float		
ReduceMax	float		
Relu	float		
Reshape			
Resize	float		**mode:** nearest / bilinear input: 4dims tensor
Resize V11	float		**mode:** nearest / bilinear exclude_outside: 0 extrapolation_value: 0 nearest_mode: floor coordinate_transformation_mode: asymmetric, align_corners(bilinear only) input: 4dims tensor
Round	float		Not supported in QuantByPass.
ScatterND (V11)	float		Only when it can be legalized using Unpool.
Shape			
Sin	float		Not supported in QuantByPass.
Slice			axis: no batch dim split step: Constant steps only. When step is not 1, the data to be sliced must be constant.
Sigmoid	float		
Softmax	float		16b Softmax can be performed along depth, width, and height and not along batch dimension. 8b Softmax can be performed only along depth.
Softplus	float		Not supported in QuantByPass.
SpaceToDepth	float		Some constraints. Blocksize must be less than UINT8.
Split	float, int32, int64		axis: no batch dim split
Squeeze			
Sqrt	float		
Sub	float		
Sum	float		

Operator	Inputs	Outputs	Restrictions
Tanh	float		
Tile	float		
TopK	float		axis: must be last dimension Must be followed by Gather Values: Not supported K must be constant
Transpose			perm: {0,2,3,1} or {0,3,1,2}
Upsample	float		mode: nearest only input: 4dims tensor
Unsqueeze			
Where			x,y: float Condition: bool

Disclaimer

免责声明

These materials are provided “as is”, except to the extent included in the selling terms and conditions of relevant products, SemiDrive® or its suppliers does not give any express or implied warranty of any kind in relation to these materials, including warranties of merchantability, title, noninfringement of intellectual property, including patents, copyrights or otherwise, or fitness for any particular purpose. SemiDrive® or its suppliers does not give any warranty as to the accuracy or completeness of the information, text, graphics, links or other items contained within these materials.

本材料按现状提供，除相关产品的销售条款与条件中列明的之外，芯驰®或其供应商并未就该等材料做任何明示或暗示的任何类型的保证，包括商品适销性、所有权、不侵犯知识产权或适用于任何特定目的。芯驰®或其供应商也不保证本资料中所包含的信息、文本、图片、链接或其他内容的完整性和准确性。

In no event will SemiDrive® or its suppliers be liable for any consequential, special, incidental, indirect, or punitive, damages, including damages for loss of profits or confidential or other information, business interruption, personal injury, property damage, loss of privacy, failure to meet any duty of good faith or reasonable care, negligence, and for any other pecuniary or other loss whatsoever, arising out of, based on, resulting from or in any way related to these materials, even if SemiDrive® or any supplier have been advised of the possibility of such damage.

在任何情况下，芯驰®或其供应商均不对任何后果性、特殊、偶然、间接或惩罚性损害负责，包括利润损失、保密信息或其他信息丢失、业务中断、人身伤害、财产损失、隐私损失、未能履行任何诚信或合理注意义务、疏忽以及因本材料引起、基于、导致或以任何方式与之相关的任何其他金钱或其他损失，即使芯驰®或任何供应商已被告知此类损害的可能性

SemiDrive® may make changes to these materials or to the products described therein, at any time without notice, but makes no commitment to making any changes. Please do not make final designs based on these materials.

芯驰®可能对本材料或其中的产品信息在不发通知的情况下更改，但不承诺作出更改，请不要基于这些材料确定最终设计。