MIOpen Porting Guide

VERSION 1.0

JULY, 2017

Key differences between MIOpen v1.0 and cuDNN:

- MIOpen only supports 4-D tensors in the NCHW storage format. This means all the "*Nd*" APIs in cuDNN do not have a corresponding API in MIOpen.
- MIOpen only supports **float** (**fp32**) data-type.
- MIOpen only supports 2D Convolutions and 2D Pooling.
- Calling miopenFindConvolution*Algorithm() is mandatory before calling any Convolution API.
- Typical calling sequence for Convolution APIs for MIOpen is:
 - miopenConvolution*GetWorkSpaceSize() // returns the workspace size required by Find()
 - o miopenFindConvolution*Algorithm() // returns performance info about various algorithms
 - o miopenConvolution*()
- MIOpen does not support Preferences for convolutions.
- MIOpen does not support Softmax modes. MIOpen implements the **SOFTMAX MODE CHANNEL** flavor.
- MIOpen does not support **Transform-Tensor**, **Dropout**, **RNNs**, and **Divisive Normalization**.

Helpful MIOpen Environment Variables

MIOPEN_ENABLE_LOGGING=1 – log all the MIOpen APIs called including the parameters passed to those APIs.

 ${\tt MIOPEN_DEBUG_AMD_ROCM_PRECOMPILED_BINARIES=0-disable~Winograd~convolution~algorithm.}$

MIOPEN_DEBUG_GCN_ASM_KERNELS=0 — disable hand-tuned asm. kernels for Direct convolution algorithm. Fall-back to kernels written in high-level language.

MIOPEN DEBUG CONV FFT=0 – disable FFT convolution algorithm.

MIOPEN DEBUG CONV DIRECT=0 - disable Direct convolution algorithm.

Operation	cuDNN API	MIOpen API
<u> </u>	cudnnStatus t	miopenStatus t
	<pre>cudnnCreate(cudnnHandle_t *handle)</pre>	<pre>miopenCreate (miopenHandle_t</pre>
	cudnnStatus t	miopenStatus t
	<pre>cudnnDestroy(cudnnHandle_t handle)</pre>	<pre>miopenDestroy (miopenHandle_t handle)</pre>
Handle	cudnnStatus_t	miopenStatus_t
пание	<pre>cudnnSetStream(cudnnHandle_t</pre>	<pre>miopenSetStream(miopenHandle_t</pre>
	handle, cudaStream_t streamId)	handle, miopenAcceleratorQueue_t streamID)
	cudnnStatus t	miopenStatus t
	cudnnGetStream(cudnnHandle t	miopenGetStream (miopenHandle t
	handle, cudaStream t *streamId)	handle, miopenAcceleratorQueue t
	manare, educationing Streaming)	*streamID)
	cudnnStatus_t	miopenStatus_t
	<pre>cudnnCreateTensorDescriptor(cudnnT</pre>	miopenCreateTensorDescriptor (mior
	<pre>ensorDescriptor_t *tensorDesc)</pre>	enTensorDescriptor_t *tensorDesc;
	cudnnStatus_t	// Only NCHW format is supported
	<pre>cudnnSetTensor4dDescriptor(cudnnTe</pre>	miopenStatus t
	nsorDescriptor_t tensorDesc,	miopenSet4dTensorDescriptor(
	<pre>cudnnTensorFormat_t format,</pre>	miopenTensorDescriptor t
	cudnnDataType_t dataType,	tensorDesc,
	int n, int c, int h, int w)	miopenDataType t dataType,
		int n, int c, int h, int w)
	cudnnStatus t	miopenStatus t
	<pre>cudnnGetTensor4dDescriptor(cudnnTe</pre>	miopenGet4dTensorDescriptor(
	nsorDescriptor t tensorDesc,	miopenTensorDescriptor t
	cudnnDataType t *dataType,	tensorDesc,
	int *n, int *c, int *h, int *w,	<pre>miopenDataType t *dataType,</pre>
	<pre>int *nStride, int *cStride,</pre>	int *n, int *c, int *h, int *w,
	<pre>int *hStride, int *wStride)</pre>	int *nStride, int *cStride,
		<pre>int *hStride, int *wStride)</pre>
_	cudnnStatus t	miopenStatus t
Tensor	cudnnDestroyTensorDescriptor (cudnn	miopenDestroyTensorDescriptor (mi
	TensorDescriptor_t tensorDesc)	penTensorDescriptor_t tensorDesc
	cudnnStatus t	// Set tensorOp to miopenOpTensorAdd
	cudnnAddTensor (cudnnHandle t	miopenStatus t
	handle,	miopenOpTensor (miopenHandle t
	const void *alpha,	handle,
	const cudnnTensorDescriptor t	miopenTensorOp t tensorOp,
	aDesc, const void *A,	const void *alpha1,
	const void *beta,	const miopenTensorDescriptor t
	const cudnnTensorDescriptor t	aDesc, const void *A,
	cDesc, void *C)	const void *alpha2,
		const miopenTensorDescriptor t
		bDesc, const void *B,
		const void *beta,
		const miopenTensorDescriptor t
		cDesc, void *C)
		// For Forward Bias use,
		miopenConvolutionForwardBias

Operation	cuDNN API	MIOpen API
Tensor	cudnnStatus_t cudnnOpTensor(cudnnHandle_t handle, const cudnnOpTensorDescriptor_t opTensorDesc, const void *alpha1, const cudnnTensorDescriptor_t aDesc, const void *A, const void *alpha2, const cudnnTensorDescriptor_t bDesc, const void *B, const void *beta, const cudnnTensorDescriptor_t cDesc, void *C)	miopenStatus_t miopenOpTensor(miopenHandle_t handle, miopenTensorOp_t tensorOp, const void *alpha1, const miopenTensorDescriptor_t aDesc, const void *A, const void *alpha2, const miopenTensorDescriptor_t bDesc, const void *B, const void *beta, const miopenTensorDescriptor_t cDesc, void *C)
	<pre>cudnnStatus_t cudnnSetTensor(cudnnHandle_t handle, const cudnnTensorDescriptor_t yDesc, void *y, const void *valuePtr)</pre>	miopenStatus_t miopenSetTensor(miopenHandle_t handle, const miopenTensorDescriptor_t yDesc, void *y, const void *alpha)
	<pre>cudnnStatus_t cudnnScaleTensor(cudnnHandle_t handle, const cudnnTensorDescriptor_t yDesc, void *y, const void *alpha)</pre>	<pre>miopenStatus_t miopenScaleTensor(miopenHandle_t handle, const miopenTensorDescriptor_t yDesc, void *y, const void *alpha)</pre>
Filter	<pre>cudnnStatus_t cudnnCreateFilterDescriptor(cudnnFilterDescriptor_t *filterDesc)</pre>	// All *FilterDescriptor* APIs are substituted by the respective TensorDescriptor APIs
Convolution	<pre>cudnnStatus_t cudnnCreateConvolutionDescriptor(c udnnConvolutionDescriptor_t *convDesc)</pre>	<pre>miopenStatus_t miopenCreateConvolutionDescriptor (miopenConvolutionDescriptor_t *convDesc)</pre>

Operation	cuDNN API	MIOpen API
	cudnnStatus_t	miopenStatus_t
	<pre>cudnnSetConvolution2dDescriptor(cu</pre>	miopenInitConvolutionDescriptor(m
	dnnConvolutionDescriptor_t	iopenConvolutionDescriptor_t
	convDesc,	convDesc,
	<pre>int pad_h, int pad_w,</pre>	miopenConvolutionMode_t mode,
	int u, int v,	int pad_h, int pad_w,
	int upscalex, int upscaley	int u, int v,
	<pre>cudnnConvolutionMode_t mode)</pre>	int upscalex, int upscaley)
	cudnnStatus_t	miopenStatus_t
	<pre>cudnnGetConvolution2dDescriptor(co</pre>	miopenGetConvolutionDescriptor(mi
	<pre>nst cudnnConvolutionDescriptor_t</pre>	openConvolutionDescriptor_t
	convDesc,	convDesc,
	<pre>int *pad_h, int *pad_y,</pre>	<pre>miopenConvolutionMode_t *mode,</pre>
	int *u, int *v,	int *pad_h, int *pad_y,
	int *upscalex, int *upscaley,	int *u, int *v,
	<pre>cudnnConvolutionMode_t *mode)</pre>	int *upscalex, int *upscaley)
	cudnnStatus_t	miopenStatus_t
	cudnnGetConvolution2dForwardOutput	miopenGetConvolutionForwardOutput
	Dim(const	<pre>Dim(miopenConvolutionDescriptor_t</pre>
	cudnnConvolutionDescriptor_t	convDesc,
	convDesc,	const miopenTensorDescriptor_t
	const cudnnTensorDescriptor_t	inputTensorDesc,
	inputTensorDesc,	const miopenTensorDescriptor_t
	<pre>const cudnnFilterDescriptor_t</pre>	filterDesc,
	filterDesc,	int *n, int *c, int *h, int *w)
	int *n, int *c, int *h, int *w)	
	cudnnStatus_t	miopenStatus_t
Convolution	cudnnDestroyConvolutionDescriptor(miopenDestroyConvolutionDescripto
	cudnnConvolutionDescriptor_t	r(miopenConvolutionDescriptor_t
	convDesc)	convDesc)
	cudnnStatus_t	<pre>// FindConvolution() is mandatory</pre>
	cudnnFindConvolutionForwardAlgorit	// Allocate workspace prior to running this API
	<pre>hm(cudnnHandle_t handle,</pre>	// A table with times and memory requirements
	const cudnnTensorDescriptor_t	for different algorithms is returned
	xDesc,	// Users can chose the top-most algorithm if they
	const cudnnFilterDescriptor_t	only care about the fastest algorithm
	wDesc, const cudnnConvolutionDescriptor t	miopenStatus_t
	convDesc,	miopenFindConvolutionForwardAlgor
	const cudnnTensorDescriptor t	<pre>ithm(miopenHandle_t handle,</pre>
	yDesc,	const miopenTensorDescriptor_t
	const int requestedAlgoCount,	xDesc,
	int *returnedAlgoCount,	const void *x,
	cudnnConvolutionFwdAlgoPerf t	const miopenTensorDescriptor_t
	*perfResults)	wDesc,
	perinebules	const void *w,
	cudnnStatus t	const
	cudnnFindConvolutionForwardAlgorit	miopenConvolutionDescriptor_t
	hmEx(cudnnHandle t handle,	convDesc,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor_t
	xDesc,	yDesc,
	const void *x,	void *y,
	const cudnnFilterDescriptor t	const int requestAlgoCount,
	wDesc,	int *returnedAlgoCount,
	· ·	miopenConvAlgoPerf_t
1		*perfResults,

Operation	cuDNN API	MIOpen API
Convolution	const void *w, const cudnnConvolutionDescriptor_t convDesc, const cudnnTensorDescriptor_t yDesc, void *y, const int requestedAlgoCount, int *returnedAlgoCount, cudnnConvolutionFwdAlgoPerf_t *perfResults, void *workSpace, size_t workSpaceSizeInBytes) cudnnStatus_t cudnnGetConvolutionForwardAlgorit hm(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const cudnnTensorDescriptor_t wDesc, const cudnnTensorDescriptor_t convDesc, const cudnnTensorDescriptor_t yDesc, cudnnConvolutionFwdPreference_t preference, size_t memoryLimitInBytes, cudnnConvolutionFwdAlgo_t *algo) cudnnStatus_t cudnnGetConvolutionForwardWorksp aceSize(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const cudnnTensorDescriptor_t xDesc, const cudnnTensorDescriptor_t wDesc, const cudnnFilterDescriptor_t wDesc, const cudnnForwardWorksp const cudnnConvolutionDescriptor_t convDesc, const cudnnTensorDescriptor_t convDesc, const cudnnTensorDescriptor_t	<pre>miopenStatus_t miopenConvolutionForwardGetWorkSpa ceSize (miopenTensorDescriptor_t wDesc, const miopenTensorDescriptor_t xDesc, const miopenConvolutionDescriptor_t convDesc, const miopenTensorDescriptor_t convDesc, const miopenTensorDescriptor_t convDesc, const miopenTensorDescriptor_t convDesc, const miopenTensorDescriptor_t</pre>
Convolution	yDesc, cudnnConvolutionFwdPreference_t preference, size_t memoryLimitInBytes, cudnnConvolutionFwdAlgo_t *algo) cudnnStatus_t cudnnGetConvolutionForwardWorksp aceSize(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const cudnnFilterDescriptor_t wDesc, const cudnnConvolutionDescriptor_t convDesc, const cudnnTensorDescriptor_t yDesc, cudnnConvolutionFwdAlgo_t algo,	<pre>miopenConvolutionForwardGetWorkSpa ceSize(miopenHandle_t handle, const miopenTensorDescriptor_t wDesc, const miopenTensorDescriptor_t xDesc, const miopenConvolutionDescriptor_t convDesc,</pre>
	cudnnStatus_t cudnnConvolutionForward(cudnnHan dle_t handle, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x,	miopenStatus_t miopenConvolutionForward(miopenHan dle_t handle, const void *alpha, const miopenTensorDescriptor_t xDesc, const void *x,

Operation	cuDNN API	MIOpen API
	const cudnnFilterDescriptor t	const miopenTensorDescriptor t
	wDesc,	wDesc,
	const void *w,	const void *w,
	const	const
	cudnnConvolutionDescriptor t	miopenConvolutionDescriptor t
	convDesc,	convDesc,
	cudnnConvolutionFwdAlgo t algo,	miopenConvFwdAlgorithm t algo,
	void *workSpace,	const void *beta,
	size t workSpaceSizeInBytes,	const miopenTensorDescriptor t
	const void *beta,	yDesc,
	const cudnnTensorDescriptor t	void *y,
	yDesc,	void *workSpace,
	void *y)	size t workSpaceSize)
	cudnnStatus t	miopenStatus t
	cudnnConvolutionBackwardBias (miopenConvolutionBackwardBias (miop
	cudnnHandle t handle,	enHandle t handle,
	const void *alpha,	const void *alpha,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	dyDesc,	dyDesc,
	const void *dy,	const void *dy,
	const void *beta,	const void *beta,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	dbDesc,	dbDesc,
	void *db)	void *db)
	cudnnStatus t	<pre>// FindConvolution() is mandatory</pre>
	cudnnFindConvolutionBackwardFilt	// Allocate workspace prior to running this API
C	erAlgorithm(cudnnHandle t	// A table with times and memory requirements
Convolution	handle,	for different algorithms is returned
	const cudnnTensorDescriptor t	// Users can chose the top-most algorithm if they
	xDesc,	only care about the fastest algorithm
	<pre>const cudnnTensorDescriptor_t</pre>	only care about the rastest algorithm
	dyDesc,	miopenStatus t
	const	miopenFindConvolutionBackwardWeigh
	cudnnConvolutionDescriptor_t	tsAlgorithm (miopenHandle t handle,
	convDesc,	const miopenTensorDescriptor t
	const cudnnFilterDescriptor_t	dyDesc,
	dwDesc,	const void *dy,
	const int requestedAlgoCount,	const word dy,
	<pre>int *returnedAlgoCount,</pre>	xDesc,
	cudnnConvolutionBwdFilterAlgoPer	const void *x,
	<pre>f_t *perfResults)</pre>	const
		miopenConvolutionDescriptor t
	cudnnStatus_t	convDesc,
	cudnnFindConvolutionBackwardFilt	const miopenTensorDescriptor t
	erAlgorithmEx(cudnnHandle_t	dwDesc,
	handle,	void *dw,
	<pre>const cudnnTensorDescriptor_t</pre>	const int requestAlgoCount,
	xDesc,	int *returnedAlgoCount,
	const void *x,	miopenConvAlgoPerf t
	const cudnnTensorDescriptor_t	*perfResults,
	dyDesc,	void *workSpace,
	const void *y,	size t workSpaceSize,
	const	bool exhaustiveSearch)
		<u>'</u>

Operation	cuDNN API	MIOpen API
Convolution	cudnnConvolutionDescriptor_t convDesc, const cudnnFilterDescriptor_t dwDesc, void *dw, const int requestedAlgoCount, int *returnedAlgoCount, cudnnConvolutionBwdFilterAlgoPer f_t *perfResults, void *workSpace, size_t workSpaceSizeInBytes) cudnnStatus_t cudnnGetConvolutionBackwardFilte rAlgorithm(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const cudnnTensorDescriptor_t dyDesc, const cudnnConvolutionDescriptor_t dwDesc, const cudnnFilterDescriptor_t dwDesc, cudnnConvolutionBwdFilterPrefere nce_t preference, size_t memoryLimitInBytes, cudnnConvolutionBwdFilterAlgo_t *algo) cudnnStatus_t cudnnGetConvolutionBackwardFilte rWorkspaceSize(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const cudnnTensorDescriptor_t	miopenStatus_t miopenConvolutionBackwardWeightsGe tWorkSpaceSize(miopenHandle_t handle, const miopenTensorDescriptor_t dyDesc, const miopenTensorDescriptor_t
	<pre>dyDesc, const cudnnConvolutionDescriptor_t convDesc, const cudnnFilterDescriptor_t gradDesc, cudnnConvolutionBwdFilterAlgo_t algo, size t *sizeInBytes)</pre>	<pre>xDesc, const miopenConvolutionDescriptor_t convDesc, const miopenTensorDescriptor_t dwDesc, size_t *workSpaceSize)</pre>

Operation	cuDNN API	MIOpen API
	cudnnStatus_t	miopenStatus_t
	cudnnConvolutionBackwardFilter(miopenConvolutionBackwardWeights(
	cudnnHandle_t handle,	miopenHandle_t handle,
	const void *alpha,	const void *alpha,
	const cudnnTensorDescriptor_t	const miopenTensorDescriptor_t
	xDesc,	dyDesc,
	const void *x,	const void *dy,
	const cudnnTensorDescriptor_t	const miopenTensorDescriptor_t
	dyDesc,	xDesc,
	const void *dy,	const void *x,
	const	const
	cudnnConvolutionDescriptor_t	miopenConvolutionDescriptor_t
	convDesc,	convDesc,
	cudnnConvolutionBwdFilterAlgo t	miopenConvBwdWeightsAlgorithm t
	algo,	algo,
	void *workSpace,	const void *beta,
	size t workSpaceSizeInBytes,	const miopenTensorDescriptor t
	const void *beta,	dwDesc,
	const cudnnFilterDescriptor t	void *dw,
	dwDesc,	<pre>void *workSpace,</pre>
	void *dw)	size t workSpaceSize)
	cudnnStatus t	miopenStatus t
	cudnnGetConvolutionBackwardDataWo	miopenConvolutionBackwardDataGetW
	rkspaceSize(cudnnHandle t	orkSpaceSize (miopenHandle t
	handle,	handle,
	const cudnnFilterDescriptor t	const miopenTensorDescriptor t
	wDesc,	dyDesc,
Convolution	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	dyDesc,	wDesc,
	const	const
	cudnnConvolutionDescriptor t	miopenConvolutionDescriptor t
	convDesc,	convDesc,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	dxDesc,	dxDesc,
	cudnnConvolutionBwdDataAlgo t	size t *workSpaceSize)
	algo,	'
	size t *sizeInBytes)	
	cudnnStatus t	<pre>// FindConvolution() is mandatory</pre>
	cudnnFindConvolutionBackwardDataA	
	lgorithm (cudnnHandle t	// Allocate workspace prior to running this API
	handle,	// A table with times and memory requirements
	const cudnnFilterDescriptor t	for different algorithms is returned
	wDesc,	// Users can chose the top-most algorithm if they
	const cudnnTensorDescriptor t	only care about the fastest algorithm
	dyDesc,	
	const	miopenStatus_t
	cudnnConvolutionDescriptor t	miopenFindConvolutionBackwardData
	convDesc,	Algorithm (miopenHandle_t handle,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor_t
	dxDesc,	dyDesc,
	const int requestedAlgoCount,	const void *dy,
	int *returnedAlgoCount,	const miopenTensorDescriptor_t
	cudnnConvolutionBwdDataAlgoPerf t	wDesc,
	*perfResults)	const void *w,
	herryeourro)	

Operation	cuDNN API	MIOpen API
•	cudnnStatus t	const
	cudnnFindConvolutionBackwardDataA	miopenConvolutionDescriptor t
	lgorithmEx (cudnnHandle t	convDesc,
	handle,	const miopenTensorDescriptor t
	const cudnnFilterDescriptor_t	dxDesc,
	wDesc,	const void *dx,
	const void *w,	const int requestAlgoCount,
	const cudnnTensorDescriptor t	<pre>int *returnedAlgoCount,</pre>
	dyDesc,	miopenConvAlgoPerf t
	const void *dy,	*perfResults,
	const	<pre>void *workSpace,</pre>
	cudnnConvolutionDescriptor_t	size_t workSpaceSize,
	convDesc,	bool exhaustiveSearch)
	const cudnnTensorDescriptor_t	
	dxDesc,	
	void *dx,	
	const int requestedAlgoCount,	
	<pre>int *returnedAlgoCount,</pre>	
	cudnnConvolutionBwdDataAlgoPerf_t	
	*perfResults,	
	void *workSpace,	
	<pre>size_t workSpaceSizeInBytes)</pre>	
	cudnnStatus_t	
	cudnnGetConvolutionBackwardDataAl	
	<pre>gorithm(cudnnHandle_t</pre>	
Convolution	handle,	
	const cudnnFilterDescriptor_t	
	wDesc,	
	const cudnnTensorDescriptor_t	
	dyDesc,	
	const	
	cudnnConvolutionDescriptor_t	
	convDesc,	
	const cudnnTensorDescriptor_t	
	dxDesc,	
	cudnnConvolutionBwdDataPreference	
	_t preference,	
	size_t memoryLimitInBytes,	
	cudnnConvolutionBwdDataAlgo_t	
	*algo)	
	and an Ct atua t	mi anangtatua t
	cudnnStatus_t	miopenStatus_t
	cudnnConvolutionBackwardData(miopenConvolutionBackwardData(mi
	cudnnHandle_t handle,	openHandle_t handle,
	const cudnatiltorDescriptor t	const void *alpha,
	const cudnnFilterDescriptor_t	const miopenTensorDescriptor_t
	wDesc, const void *w,	<pre>dyDesc, const void *dy,</pre>
	·	<u> </u>
	const cudnnTensorDescriptor_t	<pre>const miopenTensorDescriptor_t wDesc,</pre>
	<pre>dyDesc, const void *dy,</pre>	const void *w,
	Const void "dy,	COHSC VOIG "W,

const cudnnConvolutionDescriptor_t convDesc, cudnnConvolutionDesdatAlgo_t algo, void 'workSpace, size_t workSpaceSizeInBytes, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx)	Operation	cuDNN API	MIOpen API
convDesc, cudunConvolutionBwdDataAlgo_t algo, void *workSpace, size_t workSpaceSizeInBytes, const void *beta, const udnnTensorDescriptor_t dxDesc, void *dx) workSpaceSizeInBytes, const udnnTensorDescriptor_t dxDesc, void *dx) workSpaceSize t workSpaceS	•	const cudnnConvolutionDescriptor t	•
Convolution Convo		convDesc,	miopenConvolutionDescriptor t
Convolution Convolution Convolution Const void *beta, const void *beta, const void *beta, const void *dx) const cudnnTensorDescriptor_t dxDesc, void *dx, void *worKSpaces, size t worKSpaces, size t worKSpace, size t worKSpaces, size t wo		cudnnConvolutionBwdDataAlgo t	-
Convolution void *workSpaces zeInBytes, const void *beta, const void *beta, const cudnnfensorDescriptor_t dxDesc, void *dx)		_	miopenConvBwdDataAlgorithm t
Convolution const void *beta, const void *beta, const void *dx const cudnnTensorDescriptor_t dxDesc, void *dx			_
Const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx)			_
const cudnnTensorDescriptor_t dxDesc, void *dx) cudnnStatus_t cudnnSoftmaxForward(cudnnHandle_t handle, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, const void *alpha, const cudnnTensorDescriptor_t xDesc, void *y) cudnnStatus_t cudnnSoftmaxBackward(cudnnHandle_t handle, const cudnnTensorDescriptor_t yDesc, void *y) cudnnSoftmaxBackward(cudnnHandle_t handle, cudnnSoftmaxBackward(cudnnHandle_t handle, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *alpha, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dyDesc, cudnnStatus_t cudnnSetTeolingDescriptor(cudnPoolingDescriptor_t *poolingDesc, cudnnSetTeolingDescriptor(cudnPoolingDescriptor_t poolingDesc, cudnnSetTeolingAdde_t mode, cudnNanPropagation_t int windowWidth, int windowWidth, int upad_h, int pad_h, int wintow int windowWidth, int upd_h, int pad_w, int wintow int windowWidth, int wintow int window int wintow int window int wintow	Convolution	_	· ·
dxDesc,	Convolution		_
cudnnStatus_t cudnnSoftmaxForward(cudnnHandle_t handle, cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t xDesc, void *y) cudnnSoftmaxAlgorithm_t algo, const void *alpha, const cudnnTensorDescriptor_t xDesc, void *y) cudnnSoftmaxMode_t mode, cudnnSoftmaxMackward(cudnnHandle_t handle, cudnnSoftmaxAlgorithm_t algo, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y) cudnnSoftmaxMackward(cudnnSoftmaxMackward(cudnnSoftmaxMackward(cudnnSoftmaxMackward(cudnnSoftmaxMackward(cudnnSoftmaxMackward(const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *alpha, const cudnnTensorDescriptor_t dyDesc, const void *day, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const miopenTensorDescriptor_t dyDesc, const void *dy, const void *beta, const miopenTensorDescriptor_t dyDesc, const void *dy, const void *beta, const miopenTensorDescriptor_t dyDesc, const void *dy, const void *beta, const void *dy, const void *dy, const miopenTensorDescriptor_t dyDesc, const void *dy, const void *dy, const miopenTensorDescriptor_t dyDesc, const void *dy, const woid *dy, const miopenTensorDescriptor_t dyDesc, const void *dy, const woid *			,
cudnnStatus_t cudnnSoftmaxForward(cudnnHandle_t handle, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y) cudnnSoftmaxMode_t mode, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y) cudnnStatus_t cudnnSoftmaxAlgorithm_t algo, cudnsSoftmaxAlgorithm_t algo, cudnsSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnsSoftmaxAlgorithm_t algo, cudnsSoftmaxAlgorithm_t algo, cudnsSoftmaxAlgorithm_t algo, cudnsSoftmaxAlgorithm_t algo, const woid *y, const woid *alpha, const woid *alpha, const woid *alpha, const woid *alpha, const void *y, const void *dy, const void *dy, const woid *dy, const wide *mide miopenSoftmaxAlgorithm miopenSoftmaxAlgorith		· ·	*
cudnnStatus_t cudnnSoftmaxForward(cudnnHandle_t handle, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t yDesc, void *y) cudnnSoftmaxBackward(cudnnHandle_t handle, cudnnSoftmaxAlgorithm_t algo, const void *x, const void *beta, const cudnTensorDescriptor_t yDesc, void *y) cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, const void *alpha, const cudnTensorDescriptor_t yDesc, const void *alpha, const cudnTensorDescriptor_t dyDesc, const void *dy, const cudnTensorDescriptor_t dyDesc, const void *dy, const cudnTensorDescriptor_t dyDesc, const void *dy, const cudnTensorDescriptor_t dxDesc, void *dx) cudnnStatus_t cudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolingDescriptor_t poolingDesc, cudnnNaPropagation_t maxpoolingNanOpt, int windowWidth, int		VOId "dx)	
cudnnSoftmaxForward(cudnnHandle_t handle, cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y) cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxBackward(cudnnHandle_t handle, cudnnSoftmaxBackward(cudnnHandle_t handle, cudnnSoftmaxAlgorithm_t algo, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *alpha, const void *dy, const void *beta, const void *dy, const void *beta, const void *dy, const void *beta, const void *dy, const void *dy, const void *dy, const void *dy, const void *beta, const void *dy, const void *dy, const void *dy, const void *dy, const void *beta, const void *dy, const void *dy, const void *dy, const void *dy, const void *beta, const void *dy, const void *dy			_
cudnnStatus_t cudnnSoftmaxForward(cudnnHandle_t handle, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t xbesc, const void *x, const void *beta, const cudnnTensorDescriptor_t ybesc, void *y) cudnnSoftmaxBackward(cudnnBandle_t handle, cudnnSoftmaxMode_t mode, const void *alpha, const cudnTensorDescriptor_t ybesc, const void *alpha, const cudnTensorDescriptor_t ybesc, const void *alpha, const woid *alpha, const woid *alpha, const cudnTensorDescriptor_t dyDesc, const void *alpha, const miopenTensorDescriptor_t ybesc, const void *dy, const void *beta, const void *alpha, const miopenTensorDescriptor_t dyDesc, const void *dy, const miopenTensorDescriptor_t dyDesc, const woid *dy, const miopenTensorDescriptor_t dyDesc, const woid *dy, const miopenTensorDescriptor_t dyDesc, const woid *dy, c			_
cudnnSoftmaxForward(cudnnHandle_t handle, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t xDesc, void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y) Softmax S			ze)
handle, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t xDesc, void *y) CudnnSoftmaxBackward(cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t yDesc, void *alpha, const cudnnTensorDescriptor_t yDesc, const void *alpha, const cudnTensorDescriptor_t yDesc, const void *alpha, const cudnTensorDescriptor_t dyDesc, const void *dy, const cudnTensorDescriptor_t dyDesc, const void *dy, const void *dy, const void *dy, const void *dy, const void *dx) Pooling Pooling Pooling Pooling Pooling Pooling Pooling Pooling Pooling PoolingNanOpt, int windowWidth, int window			miopenStatus_t
cudnnSoftmaxMade_t mode, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnSoftmaxMackward(miopenSoftmaxMackward(miopenHandle_t handle, const void *alpha, const wid *plosec, const void *alpha, const wid *plosec, const void *alpha, const miopenTensorDescriptor_t dyDesc, const wiod *y, const wiod *y, const wiod *y, const wiod *dy, const wiod *alpha, const wiod *x, const void *beta, const miopenTensorDescriptor_t miopenSoftmaxMackward(miopenHandle_t handle, const void *dy, const wiod *dy, const wiod *dy, const wiod *y, const wiod *x, const wiod *beta, const miopenTensorDescriptor_t miopenBactus_t dyDesc, const void *d, const wiod *x, const void *beta, const miopenTensorDescriptor_t miopenBactus_t dyDesc, const wiod *dy, const wiod *alpha, const wiod *x, const wiod *beta, const miopenTensorDescriptor_t miopenBactus_t miop		<pre>cudnnSoftmaxForward(cudnnHandle_t</pre>	miopenSoftmaxForward (miopenHandl
cudnnSoftmaxMode t mode, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y) cudnnSoftmaxBackward(cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode t mode, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *dy, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *dy, const miopenTensorDescriptor_t dyDesc, const void *dy, const woid *dy, const woid *dy, const woid *dy, const miopenTensorDescriptor_t dyDesc, const void *dy, const woid *dy, const woid *dy, const woid *dy, const miopenTensorDescriptor_t dyDesc, const void *dy, const woid *dy, const void *dy, const woid *dy const woid *dy mopenFoolingDescriptor_t		handle,	e t handle,
cudnnSoftmaxMode t mode, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y) cudnnSoftmaxBackward(cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode t mode, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *dy, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *dy, const miopenTensorDescriptor_t dyDesc, const void *dy, const woid *dy, const woid *dy, const woid *dy, const miopenTensorDescriptor_t dyDesc, const void *dy, const woid *dy, const woid *dy, const woid *dy, const miopenTensorDescriptor_t dyDesc, const void *dy, const woid *dy, const void *dy, const woid *dy const woid *dy mopenFoolingDescriptor_t		cudnnSoftmaxAlgorithm t algo,	const void *alpha,
const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y) cudnnSoftmaxBackward(cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMade_t mode, const void *dalpha, const void *dalpha, const void *dalpha, const void *alpha, const void *y, const void *dy, const void *y, const void *y, const void *y, const void *dy, const void *dy pea, const voi			
const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y) cudnnStatus_t cudnnSoftmaxMackward(cudnnSoftmaxAlgorithm_t algo, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *dalpha, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnTensorDescriptor_t dxDesc, void *dx) cudnnStatus_t cudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolingDesc) cudnnStatus_t cudnnNeolingDescriptor_t cudnnPoolingDescriptor_t poolingDesc, cudnnNeolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth,		_	
XDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y)		=	•
const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y) cudnnStatus_t cudnnSoftmaxBackward(cudnnHandle_t handle, cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t yDesc, const cudnnTensorDescriptor_t dyDesc, const void *y, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx) cudnnStatus_t cudnnTensorDescriptor(cudnnPoolingDescriptor_t *poolingDesc) cudnnStatus_t cudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t cudnnPoolingDescriptor_t poolingDesc, cudnnNanPropagation_t int windowWidth, int unindowWidth, int			
Const cudnnTensorDescriptor_t yDesc, void *y)		•	_
Softmax SoftmaxBackward(miopenStatus_t miopenHandle_t handle, const void *alpha, const wiod *alpha, const wiod *pesc, const void *y, const miopenTensorDescriptor_t dyDesc, const void *y, const miopenTensorDescriptor_t dyDesc, const void *dy, const void *beta, const woid *dy, const void *beta, const woid *dy, const void *beta, const miopenTensorDescriptor_t dxDesc, void *dx) CudnnStatus_t cudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poollingDescriptor_t *poollingDescriptor_t poollingDescriptor_t poollingDescriptor_t poollingDescriptor_t poollingDescriptor_t poollingDescriptor_t poollingDescriptor_t int windowHeight, int windowWidth, int windowWidth, int u, int v) Softmax MiopenStatus_t miopenTensorDescriptor_t wiopenStatus_t miopenTensorDescriptor(miopenStatus_t miopenStatus_t miopenStatus_t miopenStatus_t miopenStatus_t miopenStatus_t miopenStatus_t miopenStatus_t miopenStatus_t miopenStatus_t miopenPoolingDescriptor(miopenPoolingDescriptor_t poollesc, miopenPoolingDescriptor_t poollesc, miopenPoolingDescriptor_t int windowWidth, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int u, int v)			<u> </u>
Softmax Sof		_	vold *y)
Softmax CudnnSoftmaxBackward(cudnnHandle_t handle, cudnnSoftmaxBackward(cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *y, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx)			
Softmax CudnnSoftmaxBackward(CudnnHandle_t handle, CudnnSoftmaxAlgorithm_t algo, CudnnSoftmaxMode_t mode, Const void *alpha, Const void *alpha, Const void *y, Const cudnnTensorDescriptor_t Const void *y, Const cudnnTensorDescriptor_t Const void *y, Const cudnnTensorDescriptor_t Const void *y, Const miopenTensorDescriptor_t Const void *dy, Const miopenTensorDescriptor_t Const void *dy, Const void *dx, Const miopenTensorDescriptor_t Const void *dy, Const void *dy, Const void *dy, Const void *dy, Const miopenTensorDescriptor_t Const void *dy, Const			
cudnnHandle_t handle, cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode_t mode, const void *alpha, const void *alpha, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx) cudnnStatus_t cudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolingDesc) cudnnStatus_t cudnnSetPooling2dDescriptor(cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int windo		_	
cudnnSoftmaxAlgorithm_t algo, cudnnSoftmaxMode_t mode, const void *alpha, const void *alpha, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx) cudnnStatus_t cudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolingDesc) cudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t poolingDesc, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int win	Softmax	•	
cudnnSoftmaxMode_t mode, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *y, const cudnnTensorDescriptor_t dyDesc, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *dy con		<u> </u>	
const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *y, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *dy, const void *dy, const void *beta, const cudnnTensorDescriptor_t dxDesc, const cudnnTensorDescriptor_t dxDesc, void *dx) cudnnStatus_t cudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolingDesc) cudnnStatus_t cudnnStatus_t cudnnPoolingDescriptor(cudnnPoolingDescriptor(cudnnPoolingDescriptor) cudnnPoolingDescriptor(cudnnPoolingDescriptor) cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int u, int v)			const void *alpha,
reconst cudnnTensorDescriptor_t yDesc, const void *y, const void *y, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx) cudnnStatus_t cudnnCreatePoolingDescriptor(cudnPoolingDescriptor_t *poolingDesc) cudnnStatus_t cudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t poolingDesc, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int unindowWidth, int		cudnnSoftmaxMode t mode,	const miopenTensorDescriptor t
Pooling Poo		const void *alpha,	yDesc,
Pooling Poo		const cudnnTensorDescriptor t	const void *y,
<pre>Pooling Const void *y,</pre>		_	const miopenTensorDescriptor t
<pre>Pooling Pooling Const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx) CudnnStatus_t cudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolIngDesc) cudnnStatus_t cudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int u, int v)</pre> const void *dy, const void *beta, const miopenTensorDescriptor_t dxDesc, void *dx) miopenCreatePoolingDescriptor(miopenPoolingDescriptor(miopenPoolingDescriptor(miopenPoolingDescriptor) miopenPoolingMode_t mode, int windowWidth, int pad_h, int pad_w, int u, int v)			_
dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx) cudnnStatus_t cudnnPoolingDescriptor_t *poolingDesc) cudnnStatus_t cudnnPoolingDescriptor(cudnnPoolingDescriptor(cudnnPoolingDescriptor) cudnnPoolingDescriptor(cudnnPoolingDescriptor) cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int un dowWidth, int un int v) *beta, const miopenTensorDescriptor_t dxDesc, void *dx) miopenCreatePoolingDescriptor(miopenSecriptor_t miopenPoolingDescriptor_t miopenPoolingDescriptor(miopenPoolingMode_t mode, int windowWidth, int windowWidth, int pad_h, int pad_w, int u, int v)			
Pooling Const void *dy, const void *beta, const miopenTensorDescriptor_t dxDesc, void *dx) CudnnStatus_t miopenCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolingDesc) CudnnStatus_t cudnnStatus_t miopenCreatePoolingDescriptor_t *poolingDesc) CudnnStatus_t miopenStatus_t miopenCreatePoolingDescriptor_t *poolingDescriptor(cudnnPoolingDescriptor(cudnnPoolingDescriptor) CudnnPoolingDescriptor_t miopenStatus_t miopenSet2dPoolingDescriptor(cudnnPoolingDescriptor_t miopenPoolingDescriptor_t poolDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int windowWidth, int windowWidth, int pad_h, int pad_h, int pad_w, int u, int v)		<u> </u>	_
reconst cudnnTensorDescriptor_t dxDesc, void *dx) CudnnStatus_t			•
dxDesc, void *dx) cudnnStatus_t cudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolingDesc) cudnnStatus_t cudnnStatus_t cudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int u, int v) miopenStatus_t miopenPoolingDescriptor(miopenPoolingDescriptor(miopenPoolingDescriptor(miopenStatus_t mio			
<pre>roid *dx) cudnnStatus_t cudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolingDesc) cudnnStatus_t cudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int u, int v)</pre> miopenStatus_t miopenSet2dPoolingDescriptor(miopenPoolingDescriptor_t poolDesc, miopenPoolingMode_t mode, int windowWidth, int pad_h, int pad_w, int u, int v)		_	
Pooling CudnnStatus_t cudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolingDesc) cudnnStatus_t cudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int u, int v) miopenStatus_t mopenSetatus_t miopenStatus_t miopenStatus_t miopenSet2dPoolingDescriptor(miopenSet2dPoolingDescriptor(miopenPoolingDescriptor_t poolDesc, miopenPoolingMode_t mode, int windowWidth, int windowWidth, int pad_h, int pad_h, int pad_w, int u, int v)		1	Void dil
Pooling Pooling CudnnCreatePoolingDescriptor(cudnnPoolingDescriptor_t *poolingDesc) cudnnStatus_t cudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowWidth, int windowWidth, int windowWidth, int u, int v) miopenCreatePoolingDescriptor(miopenPoolingDescriptor_t miopenStatus_t miopenSet2dPoolingDescriptor(miopenPoolingDescriptor_t poolDesc, miopenPoolingMode_t mode, int windowHeight, int windowWidth, int pad_h, int pad_w, int u, int v)		· ·	mionenStatus t
Pooling Pooling CudnnPoolingDescriptor_t *poolDesc) cudnnStatus_t cudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowHeight, int windowWidth, int windowWidth, int u, int v) openPoolingDescriptor_t *poolDesc) miopenPoolingDescriptor_t poolDesc, miopenPoolingMode_t mode, int windowWidth, int windowWidth, int pad_h, int pad_w, int u, int v)			
<pre>poolingDesc) cudnnStatus_t cudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowHeight, int windowWidth, int windowWidth, int windowWidth, int u, int v)</pre> <pre> *poolDesc) miopenPoolingDescriptor_t poolDesc, miopenPoolingMode_t mode, int windowHeight, int pad_h, int pad_w, int u, int v)</pre>			
reading touch not status_t cudnn Set Pooling 2d Descriptor(cudnn Pooling Descriptor_t pooling Desc, cudnn Pooling Mode_t mode, cudnn Nan Propagation_t maxpooling Nan Opt, int window Height, int window Width, int window Width, int window Width, int u, int v) miopen Status_t miopen Set 2d Pooling Descriptor(miopen Pooling Descriptor_t pool Desc, miopen Pooling Descriptor_t pool Desc, int window Height, int window Width, int window Width, int pad_h, int pad_w, int u, int v)			
Pooling CudnnSetPooling2dDescriptor(cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowHeight, int windowWidth, int windowWidth, int u, int v) miopenSet2dPoolingDescriptor(miopenPoolingDescriptor_t poolDesc, miopenPoolingMode_t mode, int windowHeight, int pad_h, int pad_w, int u, int v)			-
CudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowHeight, int windowWidth, int windowWidth, int u, int v) miopenPoolingDescriptor_t poolDesc, miopenPoolingMode_t mode, int windowHeight, int windowWidth, int pad_h, int pad_w, int u, int v)		-	_
poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowHeight, int windowWidth, int windowWidth, int windowWidth, int u, int v)			
cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowHeight, int windowHeight, int windowWidth, int windowWidth, int windowWidth, int u, int v)	Pooling		
cudnnNanPropagation_t int windowHeight, int windowWidth, int windowWidth, int windowWidth, int windowWidth, int u, int v)	. 556		
maxpoolingNanOpt, int windowWidth, int windowHeight, int pad_h, int pad_w, int windowWidth,			
<pre>int windowHeight, int windowWidth, int u, int v)</pre>		cudnnNanPropagation_t	
<pre>int windowHeight, int windowWidth, int u, int v)</pre>		maxpoolingNanOpt,	int windowWidth,
int windowWidth, int u, int v)			int pad h, int pad w,
THE VELCECATE AUGITIU.		int verticalPadding,	· · · · · ·

Operation	cuDNN API	MIOpen API
-	int horizontalPadding,	
	int verticalStride,	
	int horizontalStride)	
	cudnnStatus t	miopenStatus t
	cudnnGetPooling2dDescriptor(miopenGet2dPoolingDescriptor(
	const cudnnPoolingDescriptor t	const miopenPoolingDescriptor t
	poolingDesc,	poolDesc,
	cudnnPoolingMode t *mode,	miopenPoolingMode t *mode,
	cudnnNanPropagation t	int *windowHeight,
	*maxpoolingNanOpt,	int *windowWidth,
	int *windowHeight,	int *pad h, int *pad w,
	int *windowWidth,	int *u, int *v)
	int *verticalPadding,	The u, the v,
	int *horizontalPadding,	
	int *verticalStride,	
	int *horizontalStride)	
	cudnnStatus_t	miopenStatus_t
	cudnnGetPooling2dForwardOutputDim(miopenGetPoolingForwardOutputDin
	const cudnnPoolingDescriptor_t	const miopenPoolingDescriptor_t
	poolingDesc,	poolDesc,
	const cudnnTensorDescriptor_t	const miopenTensorDescriptor_t
	inputTensorDesc,	tensorDesc,
	int *n, int *c, int *h, int *w)	int *n, int *c, int *h, int *w)
	cudnnStatus_t	miopenStatus_t
	<pre>cudnnDestroyPoolingDescriptor(</pre>	miopenDestroyPoolingDescriptor(
	cudnnPoolingDescriptor_t	miopenPoolingDescriptor_t
Pooling	poolingDesc)	poolDesc)
	<pre>cudnnStatus t cudnnPoolingForward(</pre>	miopenStatus t
	cudnnHandle t handle,	miopenPoolingForward(
	const cudnnPoolingDescriptor t	miopenHandle t handle,
	poolingDesc,	const miopenPoolingDescriptor t
	const void *alpha,	poolDesc,
	const cudnnTensorDescriptor t	const void *alpha,
	xDesc,	const miopenTensorDescriptor t
	const void *x,	xDesc,
	const void *beta,	const void *x, const void *beta,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	yDesc,	yDesc,
	void *y)	void *y, bool do backward,
	void y)	<pre>void *y, bool do_backward, void *workSpace,</pre>
		size t workSpaceSize)
		miopenStatus t
		miopenStatus_t miopenPoolingGetWorkSpaceSize(
		const miopenTensorDescriptor_t
		yDesc,
		size_t *workSpaceSize)
	cudnnStatus_t	miopenStatus_t
	cudnnPoolingBackward(miopenPoolingBackward(
	cudnnHandle_t handle,	miopenHandle_t handle,
	const cudnnPoolingDescriptor_t	const miopenPoolingDescriptor_t
	poolingDesc,	poolDesc,
	const void *alpha,	const void *alpha,

Operation	cuDNN API	MIOpen API
	const cudnnTensorDescriptor_t	const miopenTensorDescriptor_t
	yDesc,	yDesc,
	const void *y,	const void *y,
	const cudnnTensorDescriptor_t	const miopenTensorDescriptor_t
	dyDesc,	dyDesc,
Pooling	const void *dy,	<pre>const void *dy, const miopenTensorDescriptor t</pre>
	const cudnnTensorDescriptor_t	_
	xDesc,	xDesc,
	const void *x, const void *beta, const cudnnTensorDescriptor t	<pre>const void *x, const void *beta, const miopenTensorDescriptor t</pre>
	dxDesc.	dxDesc,
	void *dx)	void *dx, const void *workspace)
	,	
	cudnnStatus_t	miopenStatus_t
	cudnnCreateActivationDescriptor(miopenCreateActivationDescriptor(
	cudnnActivationDescriptor_t	miopenActivationDescriptor_t
	*activationDesc)	*activDesc)
	cudnnStatus_t	miopenStatus_t
	cudnnSetActivationDescriptor(miopenSetActivationDescriptor(
	cudnnActivationDescriptor_t	const
	activationDesc,	miopenActivationDescriptor_t
	cudnnActivationMode_t mode,	activDesc,
	cudnnNanPropagation_t reluNanOpt,	miopenActivationMode_t mode,
	double reluCeiling)	double activAlpha,
		double activBeta,
	and an Obstacle to	double activPower)
	cudnnStatus_t	miopenStatus_t
	cudnnGetActivationDescriptor(miopenGetActivationDescriptor(
	const cudnnActivationDescriptor_t	const
	<pre>activationDesc, cudnnActivationMode t *mode,</pre>	<pre>miopenActivationDescriptor_t activDesc,</pre>
	cudnnNanPropagation t *reluNanOpt,	miopenActivationMode t *mode,
	double* reluCeiling)	double *activAlpha,
Activation	double leidceiling)	double *activArpha, double *activBeta,
Activation		I double actividata,
		·
	audnostatus t	double *activPower)
	cudnnStatus_t	double *activPower) miopenStatus_t
	cudnnDestroyActivationDescriptor(double *activPower) miopenStatus_t miopenDestroyActivationDescriptor
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t</pre>	double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc)</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc)</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle,</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle,</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle, cudnnActivationDescriptor_t</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle, const</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle, cudnnActivationDescriptor_t activationDesc,</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle, const miopenActivationDescriptor_t</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle, cudnnActivationDescriptor_t activationDesc, const void *alpha,</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle, const miopenActivationDescriptor_t activDesc,</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle, cudnnActivationDescriptor_t activationDesc, const void *alpha, const cudnnTensorDescriptor_t</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle, const miopenActivationDescriptor_t activDesc, const void *alpha,</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle, cudnnActivationDescriptor_t activationDesc, const void *alpha, const cudnnTensorDescriptor_t xDesc,</pre>	double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle, const miopenActivationDescriptor_t activDesc, const void *alpha, const miopenTensorDescriptor_t
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle, cudnnActivationDescriptor_t activationDesc, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x,</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle, const miopenActivationDescriptor_t activDesc, const void *alpha, const miopenTensorDescriptor_t xDesc,</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle, cudnnActivationDescriptor_t activationDesc, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta,</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle, const miopenActivationDescriptor_t activDesc, const void *alpha, const miopenTensorDescriptor_t xDesc, const void *x,</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle, cudnnActivationDescriptor_t activationDesc, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle, const miopenActivationDescriptor_t activDesc, const void *alpha, const miopenTensorDescriptor_t xDesc, const void *x, const void *beta,</pre>
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle, cudnnActivationDescriptor_t activationDesc, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc,</pre>	double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle, const miopenActivationDescriptor_t activDesc, const void *alpha, const miopenTensorDescriptor_t xDesc, const void *x, const void *beta, const miopenTensorDescriptor_t
	<pre>cudnnDestroyActivationDescriptor(cudnnActivationDescriptor_t activationDesc) cudnnStatus_t cudnnActivationForward(cudnnHandle_t handle, cudnnActivationDescriptor_t activationDesc, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t</pre>	<pre>double *activPower) miopenStatus_t miopenDestroyActivationDescriptor (miopenActivationDescriptor_t activDesc) miopenStatus_t miopenActivationForward(miopenHandle_t handle, const miopenActivationDescriptor_t activDesc, const void *alpha, const miopenTensorDescriptor_t xDesc, const void *x, const void *beta,</pre>

Operation	cuDNN API	MIOpen API
•	cudnnStatus t	miopenStatus t
	cudnnActivationBackward(miopenActivationBackward(
	cudnnHandle t handle,	miopenHandle t handle,
	cudnnActivationDescriptor t	const
	activationDesc,	miopenActivationDescriptor t
	const void *alpha,	activDesc,
	const cudnnTensorDescriptor t	const void *alpha,
	yDesc,	const miopenTensorDescriptor t
	const void *y,	yDesc,
Activation	const cudnnTensorDescriptor t	const void *y,
	dyDesc,	const miopenTensorDescriptor t
	const void *dy,	dyDesc,
	const cudnnTensorDescriptor t	const void *dy,
	xDesc,	const miopenTensorDescriptor t
	const void *x,	xDesc,
	const void *beta,	const void *x, const void *beta,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	dxDesc.	_
	void *dx)	<pre>dxDesc, void *dx)</pre>
	cudnnStatus t	miopenStatus t
	cudnnCreateLRNDescriptor(miopenStatus_t miopenCreateLRNDescriptor(miopenL
	cudnnLRNDescriptor t *normDesc)	RNDescriptor t *lrnDesc)
	cudnnStatus t	miopenStatus t
	cudnnSetLRNDescriptor(miopenSetLRNDescriptor(
	<u>=</u> · · · · · · · · · · · · · · · · · · ·	
	cudnnLRNDescriptor_t normDesc,	const miopenLRNDescriptor_t
	unsigned lrnN, double lrnAlpha,	lrnDesc,
	double lrnBeta, double lrnK)	miopenLRNMode_t mode,
		unsigned lrnN, double lrnAlpha,
	and a contract	double lrnBeta, double lrnK)
	cudnnStatus_t	miopenStatus_t
	cudnnGetLRNDescriptor(miopenGetLRNDescriptor(
	cudnnLRNDescriptor_t normDesc,	const miopenLRNDescriptor_t
	unsigned* lrnN, double* lrnAlpha,	lrnDesc,
	double* lrnBeta, double* lrnK)	miopenLRNMode_t *mode,
		unsigned *lrnN, double *lrnAlpha,
LRN		double *lrnBeta, double *lrnK)
	cudnnStatus_t	miopenStatus_t
	cudnnDestroyLRNDescriptor (cudnnLRN	miopenDestroyLRNDescriptor (miopen
	Descriptor_t lrnDesc)	LRNDescriptor_t lrnDesc)
	cudnnStatus_t	<pre>miopenStatus_t miopenLRNForward(</pre>
	cudnnLRNCrossChannelForward(miopenHandle_t handle,
	cudnnHandle_t handle,	const miopenLRNDescriptor_t
	<pre>cudnnLRNDescriptor_t normDesc,</pre>	lrnDesc,
	cudnnLRNMode_t lrnMode,	const void *alpha,
	const void* alpha,	const miopenTensorDescriptor_t
	<pre>const cudnnTensorDescriptor_t</pre>	xDesc,
	xDesc,	const void *x, const void *beta,
	const void *x,	const miopenTensorDescriptor t
	const void *beta,	yDesc,
	•	<u> </u>
	const cudnnTensorDescriptor t	void *y, bool do backward,
	<pre>const cudnnTensorDescriptor_t yDesc,</pre>	<pre>void *y, bool do_backward, void *workspace)</pre>

Operation	cuDNN API	MIOpen API
LRN	cudnnStatus t	miopenStatus t miopenLRNBackward(
	cudnnLRNCrossChannelBackward(miopenHandle t handle,
	cudnnHandle t handle,	const miopenLRNDescriptor t
	cudnnLRNDescriptor t normDesc,	lrnDesc,
	cudnnLRNMode t lrnMode,	const void *alpha,
	const void* alpha,	const miopenTensorDescriptor t
	const cudnnTensorDescriptor t	yDesc,
	yDesc,	const void *y,
	const void *y,	const miopenTensorDescriptor t
	const cudnnTensorDescriptor t	dyDesc,
	dyDesc,	const void *dy,
	const void *dy,	const word dy,
	=	
	const cudnnTensorDescriptor_t	xDesc,
	xDesc,	const void *x, const void *beta,
	const void *x, const void *beta,	const miopenTensorDescriptor_t
	const cudnnTensorDescriptor_t	dxDesc,
	dxDesc,	void *dx, const void *workspace)
	void *dx)	
		miopenStatus_t
		miopenLRNGetWorkSpaceSize(
		<pre>const miopenTensorDescriptor_t</pre>
		yDesc,
		size_t *workSpaceSize)
	cudnnStatus_t	miopenStatus_t
	cudnnDeriveBNTensorDescriptor(miopenDeriveBNTensorDescriptor(
	cudnnTensorDescriptor_t	miopenTensorDescriptor_t
	derivedBnDesc,	derivedBnDesc,
	const cudnnTensorDescriptor_t	const miopenTensorDescriptor_t
	xDesc,	xDesc,
	<pre>cudnnBatchNormMode_t mode)</pre>	<pre>miopenBatchNormMode_t bn_mode)</pre>
	cudnnStatus_t	miopenStatus_t
	cudnnBatchNormalizationForwardTrai	miopenBatchNormalizationForwardTr
Batch Normalization	ning(aining(
	cudnnHandle t handle,	miopenHandle t handle,
	cudnnBatchNormMode t mode,	miopenBatchNormMode t bn mode,
	void *alpha, void *beta,	void *alpha, void *beta,
	const	const
	cudnnTensorDescriptor_t	miopenTensorDescriptor_t
	xDesc,	xDesc,
	const void *x,	const void *x,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	yDesc,	yDesc,
	void *y,	void *y,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	bnScaleBiasMeanVarDesc,	bnScaleBiasMeanVarDesc,
	void *bnScale, void *bnBias,	void *bnScale, void *bnBias,
	double exponentialAverageFactor,	
	<u> </u>	double expAvgFactor,
	void *resultRunningMean,	void *resultRunningMean,
	<pre>void *resultRunningVariance,</pre>	void *resultRunningVariance,
	double epsilon,	double epsilon,
	void *resultSaveMean,	void *resultSaveMean,
	<pre>void *resultSaveInvVariance)</pre>	<pre>void *resultSaveInvVariance)</pre>

Operation	cuDNN API	MIOpen API
•	cudnnStatus t	miopenStatus t
	cudnnnBatchNormalizationForwardInf	miopenBatchNormalizationForwardIn
	erence (cudnnHandle t handle,	ference (miopenHandle t handle,
	cudnnBatchNormMode t mode,	miopenBatchNormMode t bn mode,
	void *alpha, void *beta,	void *alpha, void *beta,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	xDesc,	xDesc,
	const void *x,	const void *x,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	yDesc,	yDesc,
	void *y,	void *y,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	bnScaleBiasMeanVarDesc,	bnScaleBiasMeanVarDesc,
	<pre>const void *bnScale, void *bnBias,</pre>	<pre>void *bnScale, void *bnBias,</pre>
	const void *estimatedMean,	void *estimatedMean,
	const void *estimatedVariance,	void *estimatedVariance,
	double epsilon)	double epsilon)
	cudnnStatus t	miopenStatus t
	cudnnBatchNormalizationBackward(miopenBatchNormalizationBackward(
	cudnnHandle t handle,	miopenHandle t handle,
Batch	cudnnBatchNormMode t mode,	miopenBatchNormMode t bn mode,
Normalization	const void *alphaDataDiff,	const void *alphaDataDiff,
Normanzacion	const void *betaDataDiff,	const void *betaDataDiff,
	const void *alphaParamDiff,	const void *alphaParamDiff,
	const void *betaParamDiff,	const void *betaParamDiff,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	xDesc,	xDesc,
	const void *x,	const void *x,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	dyDesc,	dyDesc,
	const void *dy,	const void *dy,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	dxDesc,	dxDesc,
	void *dx,	void *dx,
	const cudnnTensorDescriptor t	const miopenTensorDescriptor t
	bnScaleBiasDiffDesc,	bnScaleBiasDiffDesc,
	const void *bnScale,	const void *bnScale,
	void *resultBnScaleDiff,	void *resultBnScaleDiff,
	void *resultBnBiasDiff,	void *resultBnBiasDiff,
	double epsilon,	double epsilon,
	const void *savedMean,	const void *savedMean,
	const void 'savedIneVariance)	const void *savedInvVariance)