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()
    - miopenFindConvolution*Algorithm() // returns performance info about various algorithms
    - 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.
`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
Handle	cudnnStatus_t **cudnnCreate**(cudnnHandle_t *handle)	miopenStatus_t **miopenCreate**(miopenHandle_t *handle)
	cudnnStatus_t **cudnnDestroy**(cudnnHandle_t handle)	miopenStatus_t **miopenDestroy**(miopenHandle_t handle)
	cudnnStatus_t **cudnnSetStream**(cudnnHandle_t handle, cudaStream_t streamId)	miopenStatus_t **miopenSetStream**(miopenHandle_t handle, miopenAcceleratorQueue_t streamID)
	cudnnStatus_t **cudnnGetStream**(cudnnHandle_t handle, cudaStream_t *streamId)	miopenStatus_t **miopenGetStream**(miopenHandle_t handle, miopenAcceleratorQueue_t *streamID)
Tensor	cudnnStatus_t **cudnnCreateTensorDescriptor**(cudnnTensorDescriptor_t *tensorDesc)	miopenStatus_t **miopenCreateTensorDescriptor**(miopenTensorDescriptor_t *tensorDesc)
	cudnnStatus_t **cudnnSetTensor4dDescriptor**(cudnnTensorDescriptor_t tensorDesc, cudnnTensorFormat_t format, cudnnDataType_t dataType, int n, int c, int h, int w)	// Only NCHW format is supported miopenStatus_t **miopenSet4dTensorDescriptor**(miopenTensorDescriptor_t tensorDesc, miopenDataType_t dataType, int n, int c, int h, int w)
	cudnnStatus_t **cudnnGetTensor4dDescriptor**(cudnnTensorDescriptor_t tensorDesc, cudnnDataType_t *dataType, int *n, int *c, int *h, int *w, int *nStride, int *cStride, int *hStride, int *wStride)	miopenStatus_t **miopenGet4dTensorDescriptor**(miopenTensorDescriptor_t tensorDesc, miopenDataType_t *dataType, int *n, int *c, int *h, int *w, int *nStride, int *cStride, int *hStride, int *wStride)
	cudnnStatus_t **cudnnDestroyTensorDescriptor**(cudnnTensorDescriptor_t tensorDesc)	miopenStatus_t **miopenDestroyTensorDescriptor**(miopenTensorDescriptor_t tensorDesc)

	cudnnStatus_t **cudnnAddTensor**(cudnnHandle_t handle, const void *alpha, const cudnnTensorDescriptor_t aDesc, const void *A, const void *beta, const cudnnTensorDescriptor_t cDesc, void *C)	// Set tensorOp to miopenOpTensorAdd
		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)
		// 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)
	cudnnStatus_t **cudnnSetTensor**(cudnnHandle_t handle, const cudnnTensorDescriptor_t yDesc, void *y, const void *valuePtr)	miopenStatus_t **miopenSetTensor**(miopenHandle_t handle, const miopenTensorDescriptor_t yDesc, void *y, const void *alpha)
	cudnnStatus_t **cudnnScaleTensor**(cudnnHandle_t handle, const cudnnTensorDescriptor_t yDesc, void *y, const void *alpha)	miopenStatus_t **miopenScaleTensor**(miopenHandle_t handle, const miopenTensorDescriptor_t yDesc, void *y, const void *alpha)
Filter	cudnnStatus_t **cudnnCreateFilterDescriptor**(cudnnFilterDescriptor_t *filterDesc)	// All ***FilterDescriptor*** APIs are substituted by the respective **TensorDescriptor** APIs
Convolution	cudnnStatus_t **cudnnCreateConvolutionDescriptor**(cudnnConvolutionDescriptor_t *convDesc)	miopenStatus_t **miopenCreateConvolutionDescriptor**(miopenConvolutionDescriptor_t *convDesc)

Operation	cuDNN API	MIOpen API
Convolution	cudnnStatus_t **cudnnSetConvolution2dDescriptor**(cudnnConvolutionDescriptor_t convDesc, int pad_h, int pad_w, int u, int v, int upscalex, int upscaley cudnnConvolutionMode_t mode)	miopenStatus_t **miopenInitConvolutionDescriptor**(miopenConvolutionDescriptor_t convDesc, miopenConvolutionMode_t mode, int pad_h, int pad_w, int u, int v, int upscalex, int upscaley)
	cudnnStatus_t **cudnnGetConvolution2dDescriptor**(const cudnnConvolutionDescriptor_t convDesc, int *pad_h, int *pad_y, int *u, int *v, int *upscalex, int *upscaley, cudnnConvolutionMode_t *mode)	miopenStatus_t **miopenGetConvolutionDescriptor**(miopenConvolutionDescriptor_t convDesc, miopenConvolutionMode_t *mode, int *pad_h, int *pad_y, int *u, int *v, int *upscalex, int *upscaley)
	cudnnStatus_t **cudnnGetConvolution2dForwardOutputDim**(const cudnnConvolutionDescriptor_t convDesc, const cudnnTensorDescriptor_t inputTensorDesc, const cudnnFilterDescriptor_t filterDesc, int *n, int *c, int *h, int *w)	miopenStatus_t **miopenGetConvolutionForwardOutputDim**(miopenConvolutionDescriptor_t convDesc, const miopenTensorDescriptor_t inputTensorDesc, const miopenTensorDescriptor_t filterDesc, int *n, int *c, int *h, int *w)
	cudnnStatus_t **cudnnDestroyConvolutionDescriptor**(cudnnConvolutionDescriptor_t convDesc)	miopenStatus_t **miopenDestroyConvolutionDescriptor**(miopenConvolutionDescriptor_t convDesc)

	cudnnStatus_t **cudnnFindConvolutionForwardAlgorithm**(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const cudnnFilterDescriptor_t wDesc, const cudnnConvolutionDescriptor_t convDesc, const cudnnTensorDescriptor_t yDesc, const int requestedAlgoCount, int *returnedAlgoCount, cudnnConvolutionFwdAlgoPerf_t *perfResults)
	
	cudnnStatus_t **cudnnFindConvolutionForwardAlgorithmEx**(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const void *x, const cudnnFilterDescriptor_t wDesc,
		// FindConvolution() is mandatory
		// Allocate workspace prior to running this API
		// A table with times and memory requirements for different algorithms is returned
		// Users can chose the top-most algorithm if they only care about the fastest algorithm
		miopenStatus_t **miopenFindConvolutionForwardAlgorithm**(miopenHandle_t handle, const miopenTensorDescriptor_t xDesc, const void *x, const miopenTensorDescriptor_t wDesc, const void *w, const miopenConvolutionDescriptor_t convDesc, const miopenTensorDescriptor_t yDesc, void *y, const int requestAlgoCount, int *returnedAlgoCount, miopenConvAlgoPerf_t *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 **cudnnGetConvolutionForwardAlgorithm**(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const cudnnFilterDescriptor_t wDesc, const cudnnConvolutionDescriptor_t convDesc, const cudnnTensorDescriptor_t yDesc, cudnnConvolutionFwdPreference_t preference, size_t memoryLimitInBytes, cudnnConvolutionFwdAlgo_t *algo)	void *workSpace, size_t workSpaceSize, bool exhaustiveSearch)
	cudnnStatus_t **cudnnGetConvolutionForwardWorkspaceSize**(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const cudnnFilterDescriptor_t wDesc, const cudnnConvolutionDescriptor_t convDesc, const cudnnTensorDescriptor_t yDesc, cudnnConvolutionFwdAlgo_t algo, size_t *sizeInBytes)	miopenStatus_t **miopenConvolutionForwardGetWorkSpaceSize**(miopenHandle_t handle, const miopenTensorDescriptor_t wDesc, const miopenTensorDescriptor_t xDesc, const miopenConvolutionDescriptor_t convDesc, const miopenTensorDescriptor_t yDesc, size_t *workSpaceSize)
	cudnnStatus_t **cudnnConvolutionForward**(cudnnHandle_t handle, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x,	miopenStatus_t **miopenConvolutionForward**(miopenHandle_t handle, const void *alpha, const miopenTensorDescriptor_t xDesc, const void *x,

Operation	cuDNN API	MIOpen API
Convolution	const cudnnFilterDescriptor_t wDesc, const void *w, const cudnnConvolutionDescriptor_t convDesc, cudnnConvolutionFwdAlgo_t algo, void *workSpace, size_t workSpaceSizeInBytes, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y)	const miopenTensorDescriptor_t wDesc, const void *w, const miopenConvolutionDescriptor_t convDesc, miopenConvFwdAlgorithm_t algo, const void *beta, const miopenTensorDescriptor_t yDesc, void *y, void *workSpace, size_t workSpaceSize)
	cudnnStatus_t **cudnnConvolutionBackwardBias**(cudnnHandle_t handle, const void *alpha, const cudnnTensorDescriptor_t dyDesc, const void *dy, const void *beta, const cudnnTensorDescriptor_t dbDesc, void *db)	miopenStatus_t **miopenConvolutionBackwardBias**(miopenHandle_t handle, const void *alpha, const miopenTensorDescriptor_t dyDesc, const void *dy, const void *beta, const miopenTensorDescriptor_t dbDesc, void *db)

	cudnnStatus_t **cudnnFindConvolutionBackwardFilterAlgorithm**(cudnnHandle_t handle,
	const cudnnTensorDescriptor_t xDesc,
	const cudnnTensorDescriptor_t dyDesc,
	const cudnnConvolutionDescriptor_t convDesc,
	const cudnnFilterDescriptor_t dwDesc,
	const int requestedAlgoCount,
	int *returnedAlgoCount,
	cudnnConvolutionBwdFilterAlgoPerf_t *perfResults)
	
	cudnnStatus_t **cudnnFindConvolutionBackwardFilterAlgorithmEx**(cudnnHandle_t handle,
	const cudnnTensorDescriptor_t xDesc,
	const void *x,
	const cudnnTensorDescriptor_t dyDesc,
	const void *y,
	const
		// FindConvolution() is mandatory
		// Allocate workspace prior to running this API
		// A table with times and memory requirements for different algorithms is returned
		// Users can chose the top-most algorithm if they only care about the fastest algorithm
		
		miopenStatus_t **miopenFindConvolutionBackwardWeightsAlgorithm**(miopenHandle_t handle,
		const miopenTensorDescriptor_t dyDesc,
		const void *dy,
		const miopenTensorDescriptor_t xDesc,
		const void *x,
		const miopenConvolutionDescriptor_t convDesc,
		const miopenTensorDescriptor_t dwDesc,
		void *dw,
		const int requestAlgoCount,
		int *returnedAlgoCount,
		miopenConvAlgoPerf_t *perfResults,
		void *workSpace,
		size_t workSpaceSize,
		bool exhaustiveSearch)

Operation	cuDNN API	MIOpen API
Convolution	cudnnConvolutionDescriptor_t convDesc, const cudnnFilterDescriptor_t dwDesc, void *dw, const int requestedAlgoCount, int *returnedAlgoCount, cudnnConvolutionBwdFilterAlgoPerf_t *perfResults, void *workSpace, size_t workSpaceSizeInBytes) cudnnStatus_t **cudnnGetConvolutionBackwardFilterAlgorithm**(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const cudnnTensorDescriptor_t dyDesc, const cudnnConvolutionDescriptor_t convDesc, const cudnnFilterDescriptor_t dwDesc, cudnnConvolutionBwdFilterPreference_t preference, size_t memoryLimitInBytes, cudnnConvolutionBwdFilterAlgo_t *algo)	
	cudnnStatus_t **cudnnGetConvolutionBackwardFilterWorkspaceSize**(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const cudnnTensorDescriptor_t dyDesc, const cudnnConvolutionDescriptor_t convDesc, const cudnnFilterDescriptor_t gradDesc, cudnnConvolutionBwdFilterAlgo_t algo, size_t *sizeInBytes)	miopenStatus_t **miopenConvolutionBackwardWeightsGetWorkSpaceSize**(miopenHandle_t handle, const miopenTensorDescriptor_t dyDesc, const miopenTensorDescriptor_t xDesc, const miopenConvolutionDescriptor_t convDesc, const miopenTensorDescriptor_t dwDesc, size_t *workSpaceSize)

Operation	cuDNN API	MIOpen API
Convolution	cudnnStatus_t **cudnnConvolutionBackwardFilter**(cudnnHandle_t handle, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnConvolutionDescriptor_t convDesc, cudnnConvolutionBwdFilterAlgo_t algo, void *workSpace, size_t workSpaceSizeInBytes, const void *beta, const cudnnFilterDescriptor_t dwDesc, void *dw)	miopenStatus_t **miopenConvolutionBackwardWeights**(miopenHandle_t handle, const void *alpha, const miopenTensorDescriptor_t dyDesc, const void *dy, const miopenTensorDescriptor_t xDesc, const void *x, const miopenConvolutionDescriptor_t convDesc, miopenConvBwdWeightsAlgorithm_t algo, const void *beta, const miopenTensorDescriptor_t dwDesc, void *dw, void *workSpace, size_t workSpaceSize)
	cudnnStatus_t **cudnnGetConvolutionBackwardDataWorkspaceSize**(cudnnHandle_t handle, const cudnnFilterDescriptor_t wDesc, const cudnnTensorDescriptor_t dyDesc, const cudnnConvolutionDescriptor_t convDesc, const cudnnTensorDescriptor_t dxDesc, cudnnConvolutionBwdDataAlgo_t algo, size_t *sizeInBytes)	miopenStatus_t **miopenConvolutionBackwardDataGetWorkSpaceSize**(miopenHandle_t handle, const miopenTensorDescriptor_t dyDesc, const miopenTensorDescriptor_t wDesc, const miopenConvolutionDescriptor_t convDesc, const miopenTensorDescriptor_t dxDesc, size_t *workSpaceSize)

	cudnnStatus_t
	cudnnFindConvolutionBackwardDataAlgorithm(cudnnHandle_t handle,
	const cudnnFilterDescriptor_t wDesc,
	const cudnnTensorDescriptor_t dyDesc,
	const cudnnConvolutionDescriptor_t convDesc,
	const cudnnTensorDescriptor_t dxDesc,
	const int requestedAlgoCount,
	int *returnedAlgoCount,
	cudnnConvolutionBwdDataAlgoPerf_t *perfResults)
		// FindConvolution() is mandatory
		// Allocate workspace prior to running this API
		// A table with times and memory requirements for different algorithms is returned
		// Users can chose the top-most algorithm if they only care about the fastest algorithm
		
		miopenStatus_t
		miopenFindConvolutionBackwardDataAlgorithm(miopenHandle_t handle,
		const miopenTensorDescriptor_t dyDesc,
		const void *dy,
		const miopenTensorDescriptor_t wDesc,
		const void *w,

Operation	cuDNN API	MIOpen API
Convolution	cudnnStatus_t **cudnnFindConvolutionBackwardDataAlgorithmEx**(cudnnHandle_t handle, const cudnnFilterDescriptor_t wDesc, const void *w, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnConvolutionDescriptor_t convDesc, const cudnnTensorDescriptor_t dxDesc, void *dx, const int requestedAlgoCount, int *returnedAlgoCount, cudnnConvolutionBwdDataAlgoPerf_t *perfResults, void *workSpace, size_t workSpaceSizeInBytes) cudnnStatus_t **cudnnGetConvolutionBackwardDataAlgorithm**(cudnnHandle_t 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)	const miopenConvolutionDescriptor_t convDesc, const miopenTensorDescriptor_t dxDesc, const void *dx, const int requestAlgoCount, int *returnedAlgoCount, miopenConvAlgoPerf_t *perfResults, void *workSpace, size_t workSpaceSize, bool exhaustiveSearch)
	cudnnStatus_t **cudnnConvolutionBackwardData**(cudnnHandle_t handle, const void *alpha, const cudnnFilterDescriptor_t wDesc, const void *w, const cudnnTensorDescriptor_t dyDesc, const void *dy,	miopenStatus_t **miopenConvolutionBackwardData**(miopenHandle_t handle, const void *alpha, const miopenTensorDescriptor_t dyDesc, const void *dy, const miopenTensorDescriptor_t wDesc, const void *w,

Operation	cuDNN API	MIOpen API
Convolution	const cudnnConvolutionDescriptor_t convDesc, cudnnConvolutionBwdDataAlgo_t algo, void *workSpace, size_t workSpaceSizeInBytes, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx)	const miopenConvolutionDescriptor_t convDesc, miopenConvBwdDataAlgorithm_t algo, const void *beta, const miopenTensorDescriptor_t dxDesc, void *dx, void *workSpace, size_t workSpaceSi ze)
Softmax	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)	miopenStatus_t **miopenSoftmaxForward**(miopenHandl e_t handle, const void *alpha, const miopenTensorDescriptor_t xDesc, const void *x, const void *beta, const miopenTensorDescriptor_t yDesc, void *y)
	cudnnStatus_t **cudnnSoftmaxBackward**(cudnnHandle_t handle, 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)	miopenStatus_t **miopenSoftmaxBackward**(miopenHandle_t handle, const void *alpha, const miopenTensorDescriptor_t yDesc, const void *y, const miopenTensorDescriptor_t dyDesc, const void *dy, const void *beta, const miopenTensorDescriptor_t dxDesc, void *dx)
Pooling	cudnnStatus_t **cudnnCreatePoolingDescriptor**(cudnnPoolingDescriptor_t *poolingDesc)	miopenStatus_t **miopenCreatePoolingDescriptor**(mi openPoolingDescriptor_t *poolDesc)
	cudnnStatus_t **cudnnSetPooling2dDescriptor**(cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t mode, cudnnNanPropagation_t maxpoolingNanOpt, int windowHeight, int windowWidth, int verticalPadding,	miopenStatus_t **miopenSet2dPoolingDescriptor**(miopenPoolingDescriptor_t poolDesc, miopenPoolingMode_t mode, int windowHeight, int windowWidth, int pad_h, int pad_w, int u, int v)

Operation	cuDNN API	MIOpen API
Pooling	int horizontalPadding, int verticalStride, int horizontalStride)	
	cudnnStatus_t **cudnnGetPooling2dDescriptor**(const cudnnPoolingDescriptor_t poolingDesc, cudnnPoolingMode_t *mode, cudnnNanPropagation_t *maxpoolingNanOpt, int *windowHeight, int *windowWidth, int *verticalPadding, int *horizontalPadding, int *verticalStride, int *horizontalStride)	miopenStatus_t **miopenGet2dPoolingDescriptor**(const miopenPoolingDescriptor_t poolDesc, miopenPoolingMode_t *mode, int *windowHeight, int *windowWidth, int *pad_h, int *pad_w, int *u, int *v)
	cudnnStatus_t **cudnnGetPooling2dForwardOutputDim**(const cudnnPoolingDescriptor_t poolingDesc, const cudnnTensorDescriptor_t inputTensorDesc, int *n, int *c, int *h, int *w)	miopenStatus_t **miopenGetPoolingForwardOutputDim**(const miopenPoolingDescriptor_t poolDesc, const miopenTensorDescriptor_t tensorDesc, int *n, int *c, int *h, int *w)
	cudnnStatus_t **cudnnDestroyPoolingDescriptor**(cudnnPoolingDescriptor_t poolingDesc)	miopenStatus_t **miopenDestroyPoolingDescriptor**(miopenPoolingDescriptor_t poolDesc)
	cudnnStatus_t **cudnnPoolingForward**(cudnnHandle_t handle, const cudnnPoolingDescriptor_t poolingDesc, const void *alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y)	miopenStatus_t **miopenPoolingForward**(miopenHandle_t handle, const miopenPoolingDescriptor_t poolDesc, const void *alpha, const miopenTensorDescriptor_t xDesc, const void *x, const void *beta, const miopenTensorDescriptor_t yDesc, void *y, bool do_backward, void *workSpace, size_t workSpaceSize)
		miopenStatus_t **miopenPoolingGetWorkSpaceSize**(const miopenTensorDescriptor_t yDesc, size_t *workSpaceSize)
	cudnnStatus_t **cudnnPoolingBackward**(cudnnHandle_t handle, const cudnnPoolingDescriptor_t poolingDesc, const void *alpha,	miopenStatus_t **miopenPoolingBackward**(miopenHandle_t handle, const miopenPoolingDescriptor_t poolDesc, const void *alpha,

Operation	cuDNN API	MIOpen API
Pooling	const cudnnTensorDescriptor_t yDesc, const void *y, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx)	const miopenTensorDescriptor_t yDesc, const void *y, const miopenTensorDescriptor_t dyDesc, const void *dy, const miopenTensorDescriptor_t xDesc, const void *x, const void *beta, const miopenTensorDescriptor_t dxDesc, void *dx, const void *workspace)
Activation	cudnnStatus_t **cudnnCreateActivationDescriptor**(cudnnActivationDescriptor_t *activationDesc)	miopenStatus_t **miopenCreateActivationDescriptor**(miopenActivationDescriptor_t *activDesc)
	cudnnStatus_t **cudnnSetActivationDescriptor**(cudnnActivationDescriptor_t activationDesc, cudnnActivationMode_t mode, cudnnNanPropagation_t reluNanOpt, double reluCeiling)	miopenStatus_t **miopenSetActivationDescriptor**(const miopenActivationDescriptor_t activDesc, miopenActivationMode_t mode, double activAlpha, double activBeta, double activPower)
	cudnnStatus_t **cudnnGetActivationDescriptor**(const cudnnActivationDescriptor_t activationDesc, cudnnActivationMode_t *mode, cudnnNanPropagation_t *reluNanOpt, double* reluCeiling)	miopenStatus_t **miopenGetActivationDescriptor**(const miopenActivationDescriptor_t activDesc, miopenActivationMode_t *mode, double *activAlpha, double *activBeta, double *activPower)
	cudnnStatus_t **cudnnDestroyActivationDescriptor**(cudnnActivationDescriptor_t activationDesc)	miopenStatus_t **miopenDestroyActivationDescriptor** (miopenActivationDescriptor_t activDesc)
	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, void *y)	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 yDesc, void *y)

Operation	cuDNN API	MIOpen API
Activation	cudnnStatus_t **cudnnActivationBackward**(cudnnHandle_t handle, cudnnActivationDescriptor_t activationDesc, const void *alpha, const cudnnTensorDescriptor_t yDesc, const void *y, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx)	miopenStatus_t **miopenActivationBackward**(miopenHandle_t handle, const miopenActivationDescriptor_t activDesc, const void *alpha, const miopenTensorDescriptor_t yDesc, const void *y, const miopenTensorDescriptor_t dyDesc, const void *dy, const miopenTensorDescriptor_t xDesc, const void *x, const void *beta, const miopenTensorDescriptor_t dxDesc, void *dx)
LRN	cudnnStatus_t **cudnnCreateLRNDescriptor**(cudnnLRNDescriptor_t *normDesc)	miopenStatus_t **miopenCreateLRNDescriptor**(miopenL RNDescriptor_t *lrnDesc)
	cudnnStatus_t **cudnnSetLRNDescriptor**(cudnnLRNDescriptor_t normDesc, unsigned lrnN, double lrnAlpha, double lrnBeta, double lrnK)	miopenStatus_t **miopenSetLRNDescriptor**(const miopenLRNDescriptor_t lrnDesc, miopenLRNMode_t mode, unsigned lrnN, double lrnAlpha, double lrnBeta, double lrnK)
	cudnnStatus_t **cudnnGetLRNDescriptor**(cudnnLRNDescriptor_t normDesc, unsigned* lrnN, double* lrnAlpha, double* lrnBeta, double* lrnK)	miopenStatus_t **miopenGetLRNDescriptor**(const miopenLRNDescriptor_t lrnDesc, miopenLRNMode_t *mode, unsigned *lrnN, double *lrnAlpha, double *lrnBeta, double *lrnK)
	cudnnStatus_t **cudnnDestroyLRNDescriptor**(cudnnLRN Descriptor_t lrnDesc)	miopenStatus_t **miopenDestroyLRNDescriptor**(miopen LRNDescriptor_t lrnDesc)
	cudnnStatus_t **cudnnLRNCrossChannelForward**(cudnnHandle_t handle, cudnnLRNDescriptor_t normDesc, cudnnLRNMode_t lrnMode, const void* alpha, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t yDesc, void *y)	miopenStatus_t **miopenLRNForward**(miopenHandle_t handle, const miopenLRNDescriptor_t lrnDesc, const void *alpha, const miopenTensorDescriptor_t xDesc, const void *x, const void *beta, const miopenTensorDescriptor_t yDesc, void *y, bool do_backward, void *workspace)

Operation	cuDNN API	MIOpen API
LRN	cudnnStatus_t **cudnnLRNCrossChannelBackward**(cudnnHandle_t handle, cudnnLRNDescriptor_t normDesc, cudnnLRNMode_t lrnMode, const void* alpha, const cudnnTensorDescriptor_t yDesc, const void *y, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnTensorDescriptor_t xDesc, const void *x, const void *beta, const cudnnTensorDescriptor_t dxDesc, void *dx)	miopenStatus_t **miopenLRNBackward**(miopenHandle_t handle, const miopenLRNDescriptor_t lrnDesc, const void *alpha, const miopenTensorDescriptor_t yDesc, const void *y, const miopenTensorDescriptor_t dyDesc, const void *dy, const miopenTensorDescriptor_t xDesc, const void *x, const void *beta, const miopenTensorDescriptor_t dxDesc, void *dx, const void *workspace)
		miopenStatus_t **miopenLRNGetWorkSpaceSize**(const miopenTensorDescriptor_t yDesc, size_t *workSpaceSize)
Batch Normalization	cudnnStatus_t **cudnnDeriveBNTensorDescriptor**(cudnnTensorDescriptor_t derivedBnDesc, const cudnnTensorDescriptor_t xDesc, cudnnBatchNormMode_t mode)	miopenStatus_t **miopenDeriveBNTensorDescriptor**(miopenTensorDescriptor_t derivedBnDesc, const miopenTensorDescriptor_t xDesc, miopenBatchNormMode_t bn_mode)
	cudnnStatus_t **cudnnBatchNormalizationForwardTraining**(cudnnHandle_t handle, cudnnBatchNormMode_t mode, void *alpha, void *beta, const cudnnTensorDescriptor_t xDesc, const void *x, const cudnnTensorDescriptor_t yDesc, void *y, const cudnnTensorDescriptor_t bnScaleBiasMeanVarDesc, void *bnScale, void *bnBias, double exponentialAverageFactor, void *resultRunningMean, void *resultRunningVariance, double epsilon, void *resultSaveMean, void *resultSaveInvVariance)	miopenStatus_t **miopenBatchNormalizationForwardTraining**(miopenHandle_t handle, miopenBatchNormMode_t bn_mode, void *alpha, void *beta, const miopenTensorDescriptor_t xDesc, const void *x, const miopenTensorDescriptor_t yDesc, void *y, const miopenTensorDescriptor_t bnScaleBiasMeanVarDesc, void *bnScale, void *bnBias, double expAvgFactor, void *resultRunningMean, void *resultRunningVariance, double epsilon, void *resultSaveMean, void *resultSaveInvVariance)

Operation	cuDNN API	MIOpen API
Batch Normalization	cudnnStatus_t **cudnnnBatchNormalizationForwardInference**(cudnnHandle_t handle, cudnnBatchNormMode_t mode, void *alpha, void *beta, const cudnnTensorDescriptor_t xDesc, const void *x, const cudnnTensorDescriptor_t yDesc, void *y, const cudnnTensorDescriptor_t bnScaleBiasMeanVarDesc, const void *bnScale, void *bnBias, const void *estimatedMean, const void *estimatedVariance, double epsilon)	miopenStatus_t **miopenBatchNormalizationForwardInference**(miopenHandle_t handle, miopenBatchNormMode_t bn_mode, void *alpha, void *beta, const miopenTensorDescriptor_t xDesc, const void *x, const miopenTensorDescriptor_t yDesc, void *y, const miopenTensorDescriptor_t bnScaleBiasMeanVarDesc, void *bnScale, void *bnBias, void *estimatedMean, void *estimatedVariance, double epsilon)
	cudnnStatus_t **cudnnBatchNormalizationBackward**(cudnnHandle_t handle, cudnnBatchNormMode_t mode, const void *alphaDataDiff, const void *betaDataDiff, const void *alphaParamDiff, const void *betaParamDiff, const cudnnTensorDescriptor_t xDesc, const void *x, const cudnnTensorDescriptor_t dyDesc, const void *dy, const cudnnTensorDescriptor_t dxDesc, void *dx, const cudnnTensorDescriptor_t bnScaleBiasDiffDesc, const void *bnScale, void *resultBnScaleDiff, void *resultBnBiasDiff, double epsilon, const void *savedMean, const void *savedInvVariance)	miopenStatus_t **miopenBatchNormalizationBackward**(miopenHandle_t handle, miopenBatchNormMode_t bn_mode, const void *alphaDataDiff, const void *betaDataDiff, const void *alphaParamDiff, const void *betaParamDiff, const miopenTensorDescriptor_t xDesc, const void *x, const miopenTensorDescriptor_t dyDesc, const void *dy, const miopenTensorDescriptor_t dxDesc, void *dx, const miopenTensorDescriptor_t bnScaleBiasDiffDesc, const void *bnScale, void *resultBnScaleDiff, void *resultBnBiasDiff, double epsilon, const void *savedMean, const void *savedInvVariance)