# 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<br>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<br>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)<br>// For Forward Bias use, miopenConvolutionForwardBias |

| Operation | cuDNN API | MIOpen API |
|---|---|---|
| Tensor | cudnnStatus_t **cudnnOpTensor**(cudnnHandle_t handle,<br>const cudnnOpTensorDescriptor_t opTensorDesc,<br>const void *alpha1,<br>const cudnnTensorDescriptor_t aDesc, const void *A,<br>const void *alpha2,<br>const cudnnTensorDescriptor_t bDesc, const void *B,<br>const void *beta,<br>const cudnnTensorDescriptor_t cDesc, void *C) | miopenStatus_t **miopenOpTensor**(miopenHandle_t handle,<br>miopenTensorOp_t tensorOp,<br>const void *alpha1,<br>const miopenTensorDescriptor_t aDesc, const void *A,<br>const void *alpha2,<br>const miopenTensorDescriptor_t bDesc, const void *B,<br>const void *beta,<br>const miopenTensorDescriptor_t cDesc, void *C) |
| | cudnnStatus_t **cudnnSetTensor**(cudnnHandle_t handle,<br>const cudnnTensorDescriptor_t yDesc, void *y,<br>const void *valuePtr) | miopenStatus_t **miopenSetTensor**(miopenHandle_t handle,<br>const miopenTensorDescriptor_t yDesc, void *y,<br>const void *alpha) |
| | cudnnStatus_t **cudnnScaleTensor**(cudnnHandle_t handle,<br>const cudnnTensorDescriptor_t yDesc, void *y,<br>const void *alpha) | miopenStatus_t **miopenScaleTensor**(miopenHandle_t handle,<br>const miopenTensorDescriptor_t yDesc, void *y,<br>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)<br><br>cudnnStatus_t **cudnnFindConvolutionForwardAlgorithmEx**(cudnnHandle_t handle, const cudnnTensorDescriptor_t xDesc, const void *x, const cudnnFilterDescriptor_t wDesc, | // FindConvolution() is mandatory<br>// Allocate workspace prior to running this API<br>// A table with times and memory requirements for different algorithms is returned<br>// Users can chose the top-most algorithm if they only care about the fastest algorithm<br>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,<br>const cudnnConvolutionDescriptor_t convDesc,<br>const cudnnTensorDescriptor_t yDesc,<br>void *y,<br>const int requestedAlgoCount,<br>int *returnedAlgoCount,<br>cudnnConvolutionFwdAlgoPerf_t *perfResults,<br>void *workSpace,<br>size_t workSpaceSizeInBytes)<br><br>cudnnStatus_t **cudnnGetConvolutionForwardAlgorithm**(cudnnHandle_t handle,<br>const cudnnTensorDescriptor_t xDesc,<br>const cudnnFilterDescriptor_t wDesc,<br>const cudnnConvolutionDescriptor_t convDesc,<br>const cudnnTensorDescriptor_t yDesc,<br>cudnnConvolutionFwdPreference_t preference,<br>size_t memoryLimitInBytes,<br>cudnnConvolutionFwdAlgo_t *algo) | void *workSpace,<br>size_t workSpaceSize,<br>bool exhaustiveSearch) |
| | cudnnStatus_t **cudnnGetConvolutionForwardWorkspaceSize**(cudnnHandle_t handle,<br>const cudnnTensorDescriptor_t xDesc,<br>const cudnnFilterDescriptor_t wDesc,<br>const cudnnConvolutionDescriptor_t convDesc,<br>const cudnnTensorDescriptor_t yDesc,<br>cudnnConvolutionFwdAlgo_t algo,<br>size_t *sizeInBytes) | miopenStatus_t **miopenConvolutionForwardGetWorkSpaceSize**(miopenHandle_t handle,<br>const miopenTensorDescriptor_t wDesc,<br>const miopenTensorDescriptor_t xDesc,<br>const miopenConvolutionDescriptor_t convDesc,<br>const miopenTensorDescriptor_t yDesc,<br>size_t *workSpaceSize) |
| | cudnnStatus_t **cudnnConvolutionForward**(cudnnHandle_t handle,<br>const void *alpha,<br>const cudnnTensorDescriptor_t xDesc,<br>const void *x, | miopenStatus_t **miopenConvolutionForward**(miopenHandle_t handle,<br>const void *alpha,<br>const miopenTensorDescriptor_t xDesc,<br>const void *x, |

| Operation | cuDNN API | MIOpen API |
|---|---|---|
| Convolution | const cudnnFilterDescriptor_t wDesc,<br>const void *w,<br>const cudnnConvolutionDescriptor_t convDesc,<br>cudnnConvolutionFwdAlgo_t algo,<br>void *workSpace,<br>size_t workSpaceSizeInBytes,<br>const void *beta,<br>const cudnnTensorDescriptor_t yDesc,<br>void *y) | const miopenTensorDescriptor_t wDesc,<br>const void *w,<br>const miopenConvolutionDescriptor_t convDesc,<br>miopenConvFwdAlgorithm_t algo,<br>const void *beta,<br>const miopenTensorDescriptor_t yDesc,<br>void *y,<br>void *workSpace,<br>size_t workSpaceSize) |
| | cudnnStatus_t **cudnnConvolutionBackwardBias**(cudnnHandle_t handle,<br>const void *alpha,<br>const cudnnTensorDescriptor_t dyDesc,<br>const void *dy,<br>const void *beta,<br>const cudnnTensorDescriptor_t dbDesc,<br>void *db) | miopenStatus_t **miopenConvolutionBackwardBias**(miopenHandle_t handle,<br>const void *alpha,<br>const miopenTensorDescriptor_t dyDesc,<br>const void *dy,<br>const void *beta,<br>const miopenTensorDescriptor_t dbDesc,<br>void *db) |
| | cudnnStatus_t **cudnnFindConvolutionBackwardFilterAlgorithm**(cudnnHandle_t handle,<br>const cudnnTensorDescriptor_t xDesc,<br>const cudnnTensorDescriptor_t dyDesc,<br>const cudnnConvolutionDescriptor_t convDesc,<br>const cudnnFilterDescriptor_t dwDesc,<br>const int requestedAlgoCount,<br>int *returnedAlgoCount,<br>cudnnConvolutionBwdFilterAlgoPerf_t *perfResults)<br><br>cudnnStatus_t **cudnnFindConvolutionBackwardFilterAlgorithmEx**(cudnnHandle_t handle,<br>const cudnnTensorDescriptor_t xDesc,<br>const void *x,<br>const cudnnTensorDescriptor_t dyDesc,<br>const void *y,<br>const | // FindConvolution() is mandatory<br>// Allocate workspace prior to running this API<br>// A table with times and memory requirements for different algorithms is returned<br>// Users can chose the top-most algorithm if they only care about the fastest algorithm<br><br>miopenStatus_t **miopenFindConvolutionBackwardWeightsAlgorithm**(miopenHandle_t handle,<br>const miopenTensorDescriptor_t dyDesc,<br>const void *dy,<br>const miopenTensorDescriptor_t xDesc,<br>const void *x,<br>const miopenConvolutionDescriptor_t convDesc,<br>const miopenTensorDescriptor_t dwDesc,<br>void *dw,<br>const int requestAlgoCount,<br>int *returnedAlgoCount,<br>miopenConvAlgoPerf_t *perfResults,<br>void *workSpace,<br>size_t workSpaceSize,<br>bool exhaustiveSearch) |

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

| Operation | cuDNN API | MIOpen API |
|---|---|---|
| **Convolution** | cudnnStatus_t<br>**cudnnConvolutionBackwardFilter**(<br>cudnnHandle_t handle,<br>const void *alpha,<br>const cudnnTensorDescriptor_t xDesc,<br>const void *x,<br>const cudnnTensorDescriptor_t dyDesc,<br>const void *dy,<br>const cudnnConvolutionDescriptor_t convDesc,<br>cudnnConvolutionBwdFilterAlgo_t algo,<br>void *workSpace,<br>size_t workSpaceSizeInBytes,<br>const void *beta,<br>const cudnnFilterDescriptor_t dwDesc,<br>void *dw) | miopenStatus_t<br>**miopenConvolutionBackwardWeights**(<br>miopenHandle_t handle,<br>const void *alpha,<br>const miopenTensorDescriptor_t dyDesc,<br>const void *dy,<br>const miopenTensorDescriptor_t xDesc,<br>const void *x,<br>const miopenConvolutionDescriptor_t convDesc,<br>miopenConvBwdWeightsAlgorithm_t algo,<br>const void *beta,<br>const miopenTensorDescriptor_t dwDesc,<br>void *dw,<br>void *workSpace,<br>size_t workSpaceSize) |
| | cudnnStatus_t<br>**cudnnGetConvolutionBackwardDataWorkspaceSize**(cudnnHandle_t handle,<br>const cudnnFilterDescriptor_t wDesc,<br>const cudnnTensorDescriptor_t dyDesc,<br>const cudnnConvolutionDescriptor_t convDesc,<br>const cudnnTensorDescriptor_t dxDesc,<br>cudnnConvolutionBwdDataAlgo_t algo,<br>size_t *sizeInBytes) | miopenStatus_t<br>**miopenConvolutionBackwardDataGetWorkSpaceSize**(miopenHandle_t handle,<br>const miopenTensorDescriptor_t dyDesc,<br>const miopenTensorDescriptor_t wDesc,<br>const miopenConvolutionDescriptor_t convDesc,<br>const miopenTensorDescriptor_t dxDesc,<br>size_t *workSpaceSize) |
| | cudnnStatus_t<br>**cudnnFindConvolutionBackwardDataAlgorithm**(cudnnHandle_t handle,<br>const cudnnFilterDescriptor_t wDesc,<br>const cudnnTensorDescriptor_t dyDesc,<br>const cudnnConvolutionDescriptor_t convDesc,<br>const cudnnTensorDescriptor_t dxDesc,<br>const int requestedAlgoCount,<br>int *returnedAlgoCount,<br>cudnnConvolutionBwdDataAlgoPerf_t *perfResults) | // FindConvolution() is mandatory<br>// Allocate workspace prior to running this API<br>// A table with times and memory requirements for different algorithms is returned<br>// Users can chose the top-most algorithm if they only care about the fastest algorithm<br><br>miopenStatus_t<br>**miopenFindConvolutionBackwardDataAlgorithm**(miopenHandle_t handle,<br>const miopenTensorDescriptor_t dyDesc,<br>const void *dy,<br>const miopenTensorDescriptor_t wDesc,<br>const void *w, |

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

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

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

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

| Operation | cuDNN API | MIOpen API |
|---|---|---|
| **Activation** | cudnnStatus_t<br>**cudnnActivationBackward**(<br>cudnnHandle_t handle,<br>cudnnActivationDescriptor_t<br>activationDesc,<br>const void *alpha,<br>const cudnnTensorDescriptor_t<br>yDesc,<br>const void *y,<br>const cudnnTensorDescriptor_t<br>dyDesc,<br>const void *dy,<br>const cudnnTensorDescriptor_t<br>xDesc,<br>const void *x,<br>const void *beta,<br>const cudnnTensorDescriptor_t<br>dxDesc,<br>void *dx) | miopenStatus_t<br>**miopenActivationBackward**(<br>miopenHandle_t handle,<br>const<br>miopenActivationDescriptor_t<br>activDesc,<br>const void *alpha,<br>const miopenTensorDescriptor_t<br>yDesc,<br>const void *y,<br>const miopenTensorDescriptor_t<br>dyDesc,<br>const void *dy,<br>const miopenTensorDescriptor_t<br>xDesc,<br>const void *x, const void *beta,<br>const miopenTensorDescriptor_t<br>dxDesc,<br>void *dx) |
| **LRN** | cudnnStatus_t<br>**cudnnCreateLRNDescriptor**(<br>cudnnLRNDescriptor_t *normDesc) | miopenStatus_t<br>**miopenCreateLRNDescriptor**(miopenL<br>RNDescriptor_t *lrnDesc) |
| | cudnnStatus_t<br>**cudnnSetLRNDescriptor**(<br>cudnnLRNDescriptor_t normDesc,<br>unsigned lrnN, double lrnAlpha,<br>double lrnBeta, double lrnK) | miopenStatus_t<br>**miopenSetLRNDescriptor**(<br>const miopenLRNDescriptor_t<br>lrnDesc,<br>miopenLRNMode_t mode,<br>unsigned lrnN, double lrnAlpha,<br>double lrnBeta, double lrnK) |
| | cudnnStatus_t<br>**cudnnGetLRNDescriptor**(<br>cudnnLRNDescriptor_t normDesc,<br>unsigned* lrnN, double* lrnAlpha,<br>double* lrnBeta, double* lrnK) | miopenStatus_t<br>**miopenGetLRNDescriptor**(<br>const miopenLRNDescriptor_t<br>lrnDesc,<br>miopenLRNMode_t *mode,<br>unsigned *lrnN, double *lrnAlpha,<br>double *lrnBeta, double *lrnK) |
| | cudnnStatus_t<br>**cudnnDestroyLRNDescriptor**(cudnnLRN<br>Descriptor_t lrnDesc) | miopenStatus_t<br>**miopenDestroyLRNDescriptor**(miopen<br>LRNDescriptor_t lrnDesc) |
| | cudnnStatus_t<br>**cudnnLRNCrossChannelForward**(<br>cudnnHandle_t handle,<br>cudnnLRNDescriptor_t normDesc,<br>cudnnLRNMode_t lrnMode,<br>const void* alpha,<br>const cudnnTensorDescriptor_t<br>xDesc,<br>const void *x,<br>const void *beta,<br>const cudnnTensorDescriptor_t<br>yDesc,<br>void *y) | miopenStatus_t **miopenLRNForward**(<br>miopenHandle_t handle,<br>const miopenLRNDescriptor_t<br>lrnDesc,<br>const void *alpha,<br>const miopenTensorDescriptor_t<br>xDesc,<br>const void *x, const void *beta,<br>const miopenTensorDescriptor_t<br>yDesc,<br>void *y, bool do_backward,<br>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<br>**cudnnnBatchNormalizationForwardInference**(cudnnHandle_t handle,<br>cudnnBatchNormMode_t mode,<br>void *alpha, void *beta,<br>const cudnnTensorDescriptor_t xDesc,<br>const void *x,<br>const cudnnTensorDescriptor_t yDesc,<br>void *y,<br>const cudnnTensorDescriptor_t bnScaleBiasMeanVarDesc,<br>const void *bnScale, void *bnBias,<br>const void *estimatedMean,<br>const void *estimatedVariance,<br>double epsilon) | miopenStatus_t<br>**miopenBatchNormalizationForwardInference**(miopenHandle_t handle,<br>miopenBatchNormMode_t bn_mode,<br>void *alpha, void *beta,<br>const miopenTensorDescriptor_t xDesc,<br>const void *x,<br>const miopenTensorDescriptor_t yDesc,<br>void *y,<br>const miopenTensorDescriptor_t bnScaleBiasMeanVarDesc,<br>void *bnScale, void *bnBias,<br>void *estimatedMean,<br>void *estimatedVariance,<br>double epsilon) |
| | cudnnStatus_t<br>**cudnnBatchNormalizationBackward**(<br>cudnnHandle_t handle,<br>cudnnBatchNormMode_t mode,<br>const void *alphaDataDiff,<br>const void *betaDataDiff,<br>const void *alphaParamDiff,<br>const void *betaParamDiff,<br>const cudnnTensorDescriptor_t xDesc,<br>const void *x,<br>const cudnnTensorDescriptor_t dyDesc,<br>const void *dy,<br>const cudnnTensorDescriptor_t dxDesc,<br>void *dx,<br>const cudnnTensorDescriptor_t bnScaleBiasDiffDesc,<br>const void *bnScale,<br>void *resultBnScaleDiff,<br>void *resultBnBiasDiff,<br>double epsilon,<br>const void *savedMean,<br>const void *savedInvVariance) | miopenStatus_t<br>**miopenBatchNormalizationBackward**(<br>miopenHandle_t handle,<br>miopenBatchNormMode_t bn_mode,<br>const void *alphaDataDiff,<br>const void *betaDataDiff,<br>const void *alphaParamDiff,<br>const void *betaParamDiff,<br>const miopenTensorDescriptor_t xDesc,<br>const void *x,<br>const miopenTensorDescriptor_t dyDesc,<br>const void *dy,<br>const miopenTensorDescriptor_t dxDesc,<br>void *dx,<br>const miopenTensorDescriptor_t bnScaleBiasDiffDesc,<br>const void *bnScale,<br>void *resultBnScaleDiff,<br>void *resultBnBiasDiff,<br>double epsilon,<br>const void *savedMean,<br>const void *savedInvVariance) |