NN Extension
Images and Tensors
OpenVX Neural Net Extension

- Convolution Neural Network topologies can be represented as OpenVX graphs
  - Layers are represented as OpenVX nodes
  - Layers connected by multi-dimensional tensors objects
  - Layer types include convolution, activation, pooling, fully-connected, soft-max
  - CNN nodes can be mixed with traditional vision nodes

- Import/Export Extension
  - Efficient handling of network Weights/Biases or complete networks

- The specification is provisional
  - Welcome feedback from the deep learning community
Main features OpenVX 1.1

- **Data type**
  - Fixed Q7.8 (16-bit integer)

- **Supported WL**
  - Overfeat
  - Alexnet
  - GoogLeNet versions
  - LSTM
  - RNN
OpenVX 1.2 features

- **Structure**
  - Tensor is core specification. NN is extension.
  - 2 variants: 8-bit only and 8-bit with 16-bit tensors.

- **Extra Data Type**
  - U8, S8 - for GEMMLOWP and Ristretto

- **Extra Supported WL**
  - Faster-RCNN
  - FCN
An Example of Convolution Neural Network
Tensor Framework

• vxTensor
  A multi dimensional mathematical object (2D is a matrix)
  It have minimum of 4 dimension. (Some vendor can have more dimensions)
    You can query the number of dimensions supported.
    \((X_CONTEXT_MAX_TENSOR_DIMENSIONS)\)
  It can also be used for other CV functions (Example: container for features)

• Merging and splitting vxTensor
  We added a notion of view. It is a ROI inside vxTensor.
  We Direct the graph edges with view. It create static merging and splitting

• Interaction with images
  We can make views of planar Images. (Can be understood as cast)
An Example of Convolution Neural Network

![Diagram of Convolution Neural Network](image-url)
Tensor Framework

- **vxTensor**
  A multi dimensional mathematical object (2D is a matrix)
  It have minimum of 4 dimension. (Some vendor can have more dimensions)
  You can query the number of dimensions supported.
  \( (X_CONTEXT_MAX_TENSOR_DIMENSIONS) \)
  It can also be used for other CV functions (Example: container for features)

- **Merging and splitting vxTensor**
  We added a notion of view. It is a ROI inside vxTensor.
  We Direct the graph edges with view. It create static merging and splitting

- **Interaction with images**
  We can make views of planar Images. (Can be understood as cast)
Practical Example: Tensor to Images

```c
vx_graph graph;
graph = vxCreateGraph(context);

void *ptr = NULL;

vx_tensor in;
vx_tensor wt_data[NUM_OF_LAYERS_ALEXNET][2] = { 0 }; 
vx_tensor bs_data[NUM_OF_LAYERS_ALEXNET][2] = { 0 }; 
vx_tensor out_data[NUM_OF_LAYERS_ALEXNET] = { 0 }; 
vx_tensor sub_out_data[NUM_OF_LAYERS_ALEXNET][2] = { 0 }; 

printf("Building MD Datas\n");

vx_size sizes[3] = { 227, 227, 3 }; 
in = vxCreateTensor(context, 3, sizes, VX_TYPE_INT16, Q78_FIXED_POINT_POSITION); 
if (raw) {
    // if patch was not processed yet - preprocess it
    vx_rectangle_t rect = { 0, 0, 227, 227 }; 
    vx_object_array inimgs = vxCreateImageObjectArrayFromTensor(in, &rect, 3, 
        1, VX_DF_IMAGE_S16); 
    preprocessOrigImageAlexnet(context, pInput, pAvrR, pAvrG, pAvrB, 
        inimgs); 
    vxReleaseObjectArray(inimgs); 
} else {
    // if received a preprocessed patch - use as is
    CopyToFromTensor(context, in, (int16_t*)pInput, -1, 
        VX_WRITE_ONLY); 
}

int out_dim = 0; 
for (int i = 0; i < NUM_OF_LAYERS_ALEXNET; i++) {
    int wt_dim[2] = { 0 }; 
    wt_size[2] = { 1, 1 }; 
    out_dim = 0; 
    for (int j = 0; j < dim max; j++) {
        if (wt_dim[1][i][j] > 0) {
            wt_dim[0]++;
            wt_size[0] = wt_dim[1][i][j];
        }
    }
```
Example: Preprocess Graph

• U8 Images to FP Data
  U8 Images need to be converted to Fixed point data before running the network.

\[ \text{Image}_{FP} = (\text{Image}_{U8} - \text{offset}) \times \text{scale} \]

Offset and scale used for training and they are constants. Sometime a Bit depth conversion will be needed as well.

```c
vx_node cnn_nodes[] = {vxChannelExtractNode(graph, image,
...  vxConvertDepthNode(graph, img1, img1s,
...  vxSubtractNode(graph, img1s, img_avgr,
...  ...

status |= vxVerifyGraph(graph);
if (status == VX_SUCCESS) {
    status = vxProcessGraph(graph);
}
```
An Example of Convolution Neural Network
Alexnet: Code Example 1

```c
// Nodes
if (graph) {
    vx_node cnn nodes[] = {
        // Layer (1) - Conv + Relu, Norm + Pool
        // Convolution - 3x227x227 --> 90x27x27
        vxConvolutionLayer(graph, in, wt data[0][0], bs data[0][0], conv params[0].padding x, conv param
        conv params[0].rounding policy, conv params[0].down scale size rounding, out data[0]),
        vxActivationLayer(graph, out data[0], relu params.function, relu params.a, relu params.b, out dat
        vxNormalizationLayer(graph, out data[0], norm params.type, norm params.normalization size, norm_
        out data[0][1]),
        vxPoolingLayer(graph, out data[0][2], VX NN POOLING MAX, pool params.pooling size x, pool params.po
        pool params.p pooling padding y, pool params.rounding, out data[0][3]),

        // Layer (2) - Conv + Relu, Norm + Pool
        // Convolution - 96x27x27 --> 256x13x13
        vxConvolutionLayer(graph, sub out data[3][0], bs data[4][0], conv params[1].padding x, conv param
        conv params[1].rounding policy, conv params[1].down scale size rounding, sub out data[4]
        vxActivationLayer(graph, sub out data[4][0], relu params.function, relu params.a, relu params.b,
        vxConvolutionLayer(graph, sub out data[4][1], bs data[4][1], conv params[1].padding x, conv param
        conv params[1].rounding policy, conv params[1].down scale size rounding, sub out data[4]
        vxActivationLayer(graph, sub out data[4][1], relu params.function, relu params.a, relu params.b,
        vxNormalizationLayer(graph, sub out data[4][1], norm params.type, norm params.normalization size, norm_
        out data[4][1]),
        vxPoolingLayer(graph, sub out data[4][2], VX NN POOLING MAX, pool params.pooling size x, pool params.po
        pool params.p pooling padding y, pool params.rounding, out data[4][3]),

        // Layer (3) - Conv + Relu
        // Convolution - 256x13x13 --> 384x13x13
        vxConvolutionLayer(graph, out data[7], wt data[8][0], bs data[8][0], conv params[2].padding x, c
        conv params[2].rounding policy, conv params[2].down scale size rounding, out data[8]),
        vxActivationLayer(graph, out data[8], relu params.function, relu params.a, relu params.b, out dat

        // Layer (4) - Conv + Relu
        // Convolution - 384x13x13 --> 384x13x13
        vxConvolutionLayer(graph, sub out data[9][0], wt data[10][0], bs data[10][0], conv params[2].pad
        conv params[2].rounding policy, conv params[2].down scale size rounding, sub out data[10]
        vxActivationLayer(graph, sub out data[10][0], relu params.function, relu params.a, relu params.b
        vxConvolutionLayer(graph, sub out data[10][1], wt data[10][1], bs data[10][1], conv params[2].pad
        conv params[2].rounding policy, conv params[2].down scale size rounding, sub out data[10]
```
Auto generation example

• Problem
  - Need to quantize trained networks from training environment.
  - Need to convert network to OpenVX

• Solution
  - Tools to convert Framework networks to OpenVX
    Example is Model optimizer from Intel
Alexnet: Auto generated code from Model Optimizer