1. Introduction
2. Related work
   1. Convolutional Neural Network (CNN)
   2. Neuromorphic Architecture

The Neuromorphic Architecture has demonstrated the hardware capable of deploying the elements, called neurons and synapses, of brain-inspired computing architectures, and the architecture has shown low-power consumption[ref9].

1. Fast Convolution Neuromorphic Processor

We proposed Fast Convolution Neuromorphic Processor Units (Fast ConvNPU) in a TDNN synthesis way proposed in [Insight]. Our proposed scheme for CNN with Consolidated Receptive Field (CRFs) to calculate output pixel values from each partial convolution windows of CRFs consistently.

Figure 1: Consolidated receptive field (CRF) which has two partial receptive fields in a row-wise direct-ion. A CRF consists of overlapped partial receptive fields which normal CNN architecture has.



Since CRF is composed of partial receptive fields, the more partial receptive fields composes CRF, the faster the CNN is able to perform convolution process.



1. TDNN Synthesis

Converting CNN into TDNN has been introduced in [INsight]. When incoming data stream is fed into TDNN reconfigured from CNN, the input nodes receive continuous pixel values and it is necessary to hold pixel values to perform convolution process. In [Insight]’s neuromorphic chip, the receptive field comprises *delay units*, *synapses*, and *neurons* to make receptive field of convolution layers and max-pooling layers in a TDNN fashion.

*Pixel-delay* unit holds pixel value for a clock cycle, and *delay lines* are composed of serialized *pixel-delay*sline by line. Each pixel values delayed by delay units *Synapse units* of the neuromorphic chip consists of multipliers and adder trees. In our scheme, we proposed following algorithm which deploys input nodes and expand them with delay-units to synthesize our Fast-CNN architecture on neuromorphic chip in a TDNN fashion.

1. Pixel List Expanding (PLE) algorithm

PLE algorithm is to expand the list of pixel nodes that CRF needs for convolution process. This list of pixel nodes is expressed as matrix. Let be the number of partial receptive fields in a row-wise direction, and let is the number of partial receptive fields in a column-wise direction.

If CRF comprises partial receptive fields, the input nodes is expressed as matrix (see Figure 2 (a)).

|  |
| --- |
| **Algorithm 1. PLE algorithm** |
| **Input:**  receptive field ,  1: the list of pixel nodes L, expressed as matrix.  2: expand column nodes of L until the column of L, is  3: expand row nodes of L until the row of L,  is |

Note that this algorithm is able to adjust not only to convolution layer, but also max-pooling layer by adding the constraints of stride of max-pooling.

To

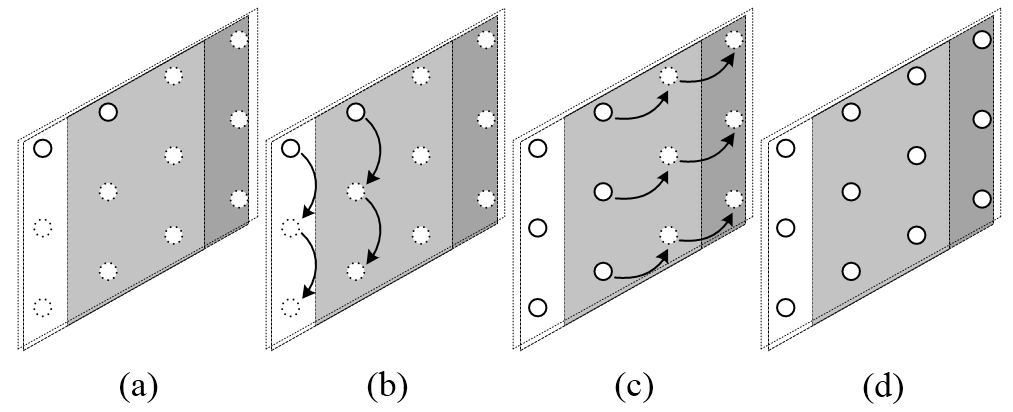


Figure 2: the case of CRF which has two partial receptive fields ( whose size is 3-by-3 in a row-wise direction. By PLE algorithm, CRF is filled with pixel nodes which CRF needs to calculate output pixel values, see (a) to (d).

1. Modified PLE algorithm: max-pooling

We modified PLE algorithm by adding the constraints. In size of max-pooling window, let s is stride, and let is the number of possible CRFs calculated by Algorithm 2 in a row direction at pooling layer and let is the number of possible CRFs calculated in a same way.

1. Resource requirement analysis

* Delay units

In convolution layer, required amount of delay units are expressed as

Let is requirements of pixel-delay, and let is requirements of delay lines. Note that goes to due to decreasing size of delay lines as .

* Neuron units

1. Experimental Results

