Deep Learning Feature Trees

This page outlines Backend Development Cost (BDC) analysis of Deep Learning accelerators. The Feature Tree Template (<FT>) used for this analysis is available below. We used the provided <FT> to model the following hardware:

- Unicore RISC-V CPU
- Intel Xeon CPU
- NVIDIA Fermi GPU
- NVIDIA Turing GPU
- Google TPU v1
- Versatile Tensor Accelerator (VTA)
- Intel Spring Hill
- NVIDIA Simba

Feature Tree Template (<FT>)

- **System Root**
  combine = SUM; *weight* = 1
  - **Memory Hierarchy**
    combine = SUM; *weight* = 1
    - **Implicit Data Movement**: No. of cache levels
      stage_mask = [0, 0, 1, 1]; *scale* = Linear; *weight* = 1
    - **Explicit Data Movement**: No. of scratchpad levels
      stage_mask = [1, 1, 1, 1]; *scale* = Linear; *weight* = 1
    - **Software Coherency**: No. of software coherrent levels
      stage_mask = [0, 1, 0, 1]; *scale* = Linear; *weight* = 1
    - **Data Access Granularity**: No. of data access granularity levels
      stage_mask = [1, 1, 1, 1]; *scale* = Linear; *weight* = 1
    - **Storage Properties**: No. of storage properties [Activations, Weights, Output]
      stage_mask = [1, 0, 1, 0]; *scale* = Linear; *weight* = 1
  - **Node Types Set**
    combine = SUM; *weight* = 1
    - **Node**
      combine = SUM; *weight* = 1
      - **Data Movement**
        combine = SUM; *weight* = 1
        - **DMA Engines**: DMA engines present in node [y/n]
          stage_mask = [0, 0, 1, 1]; *scale* = Linear; *weight* = 1
        - **Stride**: No. of dimensions of hardware supported stride
          stage_mask = [1, 0, 1, 0]; *scale* = Linear; *weight* = 1
        - **Patterns**: No. of movement patterns [Peer-to-Peer, Broadcast, Scatter]
          stage_mask = [0, 1, 0, 1]; *scale* = Linear; *weight* = 1
      - **Control**
        combine = SUM; *weight* = 1
        - **Indirect Addressing**: Node supports indirect addressing [y/n]
          stage_mask = [0, 1, 0, 0]; *scale* = Linear; *weight* = 1
        - **Loop Levels**: No. of loop levels from full control flow
          stage_mask = [0, 1, 0, 0]; *scale* = Linear; *weight* = 1
        - **Latency Hiding**: No. of latency hiding widgets [Threads, Double Buffers]
          stage_mask = [0, 0, 1, 1]; *scale* = Exponential; *weight* = 1
        - **Data Dependency**: Sync. support between data producer and consumer
          stage_mask = [1, 0, 1, 0]; *scale* = Linear; *weight* = 1
        - **ISA Specialization**: Avg no. of ISA commands per operator
          stage_mask = [1, 1, 1, 1]; *scale* = Linear; *weight* = 1

        - **Datapaths Set**
          combine = SUM; *weight* = 1
            - **DataPath**
              combine = SUM; *weight* = 1
                - **Operation Dimensions**: No. of dimensions of data operations
                  stage_mask = [1, 0, 0, 0]; *scale* = Linear; *weight* = 1
                - **Unmaskable Dimensions**: No. of inner operation dimensions without masking support
                  stage_mask = [1, 0, 0, 0]; *scale* = Linear; *weight* = 1
                - **Memory Levels**: No. of memory units in datapath [Input, Output, Internal]
                  stage_mask = [1, 0, 1, 0]; *scale* = Linear; *weight* = 1
                - **Latency Hiding**: No. of latency hiding widgets [Threads, Double Buffers]
                  stage_mask = [0, 0, 1, 0]; *scale* = Linear; *weight* = 1
    - **Network and Synchronization**
      combine = SUM; *weight* = 1
        - **Latency**: No. of latency domains visible to node
          stage_mask = [0, 0, 0, 1]; *scale* = Linear; *weight* = 1
        - **Bandwidth**: No. of bandwidth domains visible to node
          stage_mask = [0, 0, 0, 1]; *scale* = Linear; *weight* = 1
        - **Topology Positions**: No. of topology positions in node layout
          stage_mask = [0, 0, 0, 1]; *scale* = Linear; *weight* = 1
        - **Sync Capability**: No. of inter-node sync. capabilities [Atomics, Interrupt]
          stage_mask = [0, 1, 0, 1]; *scale* = Linear; *weight* = 1

<FT> Details

TODO: Add details here

Organization

The organizes Hardware Features Descriptor (HDF) into describes hardware as organizes the hardware into following components.

Network and Synchronization : Latency

TODO: Per feature details here

References

TODO: Add