Deep Learning Compiler

AWS AI
Acknowledgement
Amazon Sagemaker Neo

Enables developers to train machine learning models once and run them anywhere in the cloud and at the edge

Hardware targets
- Intel CPU, Intel graphics
- ARM CPU, ARM GPU
- Nvidia GPU
- FPGA
- ASIC
- ...

Product targets
- Amazon Rekognition
- AWS DeepLens
- Amazon Lex
- ...
- And a lot of internal/external products
CONV Kernel tuning
Intel Xeon Platinum 8000-series CPUs (Skylake)

• Multi-cores
  • E.g., EC2 c5.9xlarge: 1 processor with 18 cores.

• AVX-512 supported
  • 512-bit width registers (ZMM)
  • E.g. vfmadd231ps -1664(%rax,%r13){1to16}, %zmm0, %zmm1
CONV optimization

Data layout is **important**!

```python
cconv = tvm.compute(oshape, 
    lambda n, oc, oh, ow:
    tvm.sum(
        data[n, ic, oh*stride+kh, ow*stride+kw] * kernel[oc, ic, kh, kw],
        axis=[ic, kh, kw]),
    )
```

- NCHW -> NHWC
- NCHW -> NCHW[x]c
  - OIHW-> OIHW[x]i[y]o

for (n, 0, N):
  for (oc, 0, OC):
    for (oh, 0, OH):
      for (ow, 0, OW):
        Out[n, oc, oh, ow] = 0 // init Out
      for (ic, 0, IC):
        for (kh, 0, KH):
          for (kw, 0, KW):
            // Out += In * Kernel
CONV optimization

Utilize the AVX-512 ISA well

(broadcast) Load input to DRAM;
Load kernels to ZMM; // up to 16 float32
vfmadd input, kernel, output
Store output back to DRAM

Load 31 inputs to DRAM;
Load kernels to ZMM;
vfmadd input_1, kernel, output_1
vfmadd input_2, kernel, output_2
...
vfmadd input_31, kernel, output_31
Store output_{1…31} back to DRAM
Intel Graphics on Amazon DeepLens

Hardware Configs: Intel HD Graphics 500 (Intel’s Gen 9)

- On-die integrated GPU
- 12 EUs, 0.55 GHz
- 7 physical threads per EU, 2 128-bit FPUs per EU
- 105.6 GFLOPS peak performance
- Work items in the same SIMD group form a subgroup sharing 4KB GRFs
  - Intel Opencl ext: cl_intel_subgroups
- Shares the main memory with CPU
Instruction examples and corresponding TVM instructions

- `intel_sub_group_block_read/write` ⇒ `cache_read/write(buffer, “warp”, [result])`
- `Intel_sub_group_shuffle` ⇒ `storage_align(axis, 16)` and bind it to threads

Convolution:

- Work items work on a certain block of workloads to utilize local memory
- Layout transform for coalescing memory accesses
- Utilize `cl_intel_subgroups` operations
Graph-level optimization
Graph-level layout optimization

```
Data -> NCHW

FLATTEN
  NCHW
  OIHW

CONV
  NCHW
  OIHW

RELU
  NCHW

BATCH_NORM
  NCHW
  C

CONV
  OIHW

AlterOpLayout
```

```
Data -> NCHW

FLATTEN
  NCHW
  OIHW

CONV_NCHW16c
  NCHW
  OIHW

RELU
  NCHW

BATCH_NORM
  NCHW
  C16c

CONV_NCHW16c
  OIHW

LayoutTransform
  NCHW

Kernel

Mean / Variance
```

LayoutTransform for parameters can be pre-computed during compile time.

Optimized layout
Graph/tensor co-optimization

Dynamic programming + necessary heuristics
End-to-end results

Batch size = 1

Intel CPU

Intel Graphics
Other functionalities
Runtime multi-threading

Use a customized thread pool for CPU targets
• Lock-free queue using C++ atomics
• Thread-binding to physical cores
• Cache line padding
Graph Annotation

Annotation

Copy node insertion

Optimization/Compilation

Runtime

© 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
Quantization on Intel CPUs

Hardware support: Fast INT8 operations with INT32 accumulation

INT8 conv2d kernel requires new schedule

- Performs reduction in groups of 4 INT8 elements to INT32 elements
- FP32 schedule does not require in-vector reduction
ASICs – AWS Inherentia
Takeaways
Takeaways

• Industry needs an open standard compiler for DL
  • AWS working on the TVM stack
Takeaways

• Industry needs an open standard compiler for DL
  • AWS working on the TVM stack

• We are eager to collaborate with the community
  • Talk to us, we have 10+ people here today!
Takeaways

• Industry needs an open standard compiler for DL
  • AWS working on the TVM stack

• We are eager to collaborate with the community
  • Talk to us, we have 10+ people here today!
Takeaways

• Industry needs an open standard compiler for DL
  • AWS working on the TVM stack

• We are eager to collaborate with the community
  • Talk to us, we have 10+ people here today!

• We are hiring!
  • Write to Vin Sharma (vinarm@amazon.com) or Yida Wang (wangyida@amazon.com)