MileStone2 Report
ECE408

Team : lakersfirst
School affiliation: campus students
hz43 (Haoyuan Zhang) xiangl14 (Xiang Li) xic10 (Xi Chen)

1.List of all kernels that collectively consume more than 90% of the program time

Name	Time(%)	Time	Calls	Avg	Min	Max
[CUDA memcpy HtoD]	31.68%	35.741ms	20	1.7871ms	1.0560us	33.422ms
volta_scudnn_128x64_relu_interior_nn_v1	17.57%	19.821ms	1	19.821ms	19.821ms	19.821ms
volta_gcgemm_64x32_nt	16.99%	19.168ms	4	4.7921ms	4.7885ms	4.7966ms
fft2d_c2r_32x32	8.63%	9.7399ms	4	2.4350ms	2.0336ms	3.1590ms
volta_sgemm_128x128_tn	7.71%	8.7011ms	1	8.7011ms	8.7011ms	8.7011ms
op_generic_tensor_kernel	6.50%	7.3307ms	2	3.6653ms	25.952us	7.3047ms
fft2d_r2c_32x32	6.38%	7.1967ms	4	1.7992ms	1.4356ms	2.2607ms
cudnn::detail::pooling_fw_4d_kernel	3.90%	4.4044ms	1	4.4044ms	4.4044ms	4.4044ms

2.List of all CUDA API calls that collectively consume more than 90% of the program time

API Calls	Time	Time(%)	Calls	Avg
cudaStreamCreateWithFlags	3.68363s	43.57%	22	167.44ms
cudaMemGetInfo	2.68858s	31.80%	24	112.02ms
cudaFree	1.79272s	21.21%	19	94.354ms

3.Explanation of the difference between kernels and API calls

The kernel is the device code(run in GPU) that marked with CUDA keywords for data-parallel functions and kernels are usually written by ourselves.
API calls are calls made by the host code into the CUDA driver or runtime libraries, such as cudaMalloc(), cudaFree() and cudaMemcpy() functions.

4. Output of rai running MXNet on the CPU

Loading fashion-mnist data... done
Loading model... done
New Inference
EvalMetric: {'accuracy': 0.8154}

5. Program run time on CPU

17.10user 4.86system 0:09.07elapsed

6. Output of rai running MXNet on the GPU

Loading fashion-mnist data... done
Loading model... done
New Inference
EvalMetric: {'accuracy': 0.8154}

7. Program run time on GPU

5.11user 3.41system 0:04.72elapsed

8. CPU Program Implementation

```
void forward(mshadow::Tensor<cpu, 4, DType> &y, const mshadow::Tensor<cpu, 4, DType>
&x, const mshadow::Tensor<cpu, 4, DType> &k)
{
   const int B = x.shape_[0];
   const int M = y.shape_[1];
   const int C = x.shape_[1];
   const int H = x.shape_[2];
   const int W = x.shape_[3];
   const int K = k.shape_[3];
   int H_out = H - K + 1;
   int W_out = W - K + 1;
   for (int b = 0; b < B; ++b)                // for each image in batch
      for(int m = 0;  m < M;  m++)             // for each output feature map
         for(int h = 0; h < H_out; h++)          // for each output element
            for(int w = 0; w < W_out; w++) {
               y[b][m][h][w] = 0;
               for(int c = 0;  c < C; c++)        // sum over all input feature maps (channels)
```

```
                    for(int p = 0; p < K; p++)                // KxK  filter
                        for(int q = 0; q < K; q++)
                            y[b][m][h][w] += x[b][c][h + p][w + q] * k[m][c][p][q];
                }
}
```

9.Whole program execution time

M2.1 Output:
Loading fashion-mnist data... done
Loading model... done
New Inference
Op Time: 11.806489
Op Time: 60.063094
Correctness: 0.7653 Model: ece408
85.46user 7.61system 1:15.77elapsed 122%CPU (0avgtext+0avgdata 6044368maxresident)k
0in
puts+0outputs (0major+2310434minor)pagefaults 0swaps

Whole Program run time: User: 85.46; Sys: 7.61 Elapsed 1:15.77

10. Op Time

Op Time: 11.806489
Op Time: 60.063094